

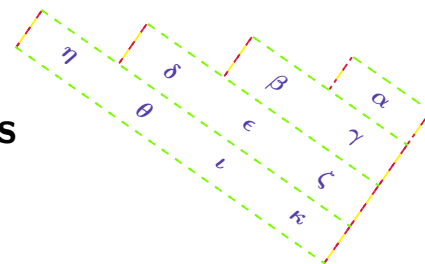
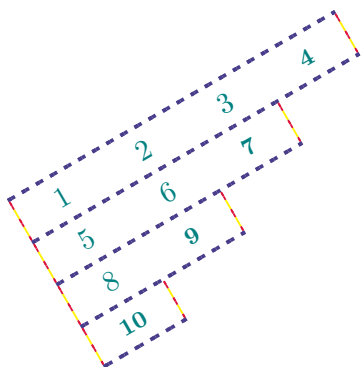


tabu and longtabu

# Flexible L<sup>A</sup>T<sub>E</sub>X tabulars

FC

2011/02/19 – version 2.5 release



## Abstract

This package defines a single environment `tabu` to make all kinds of tabulars in text or in math mode provided that they do not split across pages.

An environment `longtabu` – based on D. Carlisle `longtable` package – is also provided to make tabulars that can stretch out on several pages, while keeping some features (not all of them) of the `tabu` environment.

`tabu` is more flexible than `tabular`, `tabular*`, `tabularx` and `array` and extends the possibilities. All tabulars in this document were made with the `tabu` environment, *of course*... The implementation is optimised to minimise the measurements required to put all together.

`\tau_N b \subset` likes colors too, with special lines that are able to keep the alignment of the surrounded text... and also like numbers with the possibility to embed `\siunitx` `S` (or `s`) columns. `\tau_N b \subset` does not modify any of the macro defined by `array.sty` or in the L<sup>A</sup>T<sub>E</sub>X kernel<sup>1</sup>.

`\tau_N b \subset` requires  $\epsilon$ -T<sub>E</sub>X and the standard package `array.sty`. Natural widths of columns are computed (but not printed) by the code of `varwidth` by D. Arseneau. Finally `longtabu` is based on `longtable`.

## Contents

Summary of the features provided by <code>tabu</code> .....	3
1 Examples and counterexamples .....	5
1.1 “Locally global” settings and their scopes .....	5
1.2 X column widths computation .....	6
1.3 Inserting Verbatim material ( <code>fancyvrb</code> ) .....	7
1.4 Maths inside <code>tabu</code> X columns .....	7
1.5 Embedding <code>sunitx</code> <code>S</code> columns inside X columns .....	8
2 The <code>tabu</code> environment .....	8
2.1 <code>tabu</code> , <code>tabu to</code> and <code>tabu spread</code> .....	8
2.2 <code>longtabu</code> , <code>longtabu to</code> and <code>longtabu spread</code> .....	9
2.3 <code>tabu</code> X columns – Mastering horizontal space.....	10
2.3.1 X columns with “ <code>tabu spread</code> ” .....	
2.3.2 <i>Negativ width coefficients for X columns</i> .....	
2.3.3 <i>Multicolumn in tabu</i> .....	
2.4 <code>\tabulinesep</code> and <code>\extrarowsep</code> – Mastering vertical space.....	12
2.5 <code>tabu</code> in math mode .....	14
3 Lines leaders and colors inside <code>tabu</code> .....	14
3.1 First important remark .....	14
3.2 Vertical lines: <code> </code> has an optional parameter .....	14
3.3 Multiple <code>\firsthline</code> and <code>\lasthline</code> .....	15
3.4 More style for lines .....	16
3.5 Automatic horizontal lines and row colors .....	17
4 Modifying the font and the alignment in one row: <code>\rowfont</code> .....	18
5 Saving and restoring a <code>tabu</code> : <code>\savetabu</code> , <code>\usetabu</code> and <code>\preamble</code> .....	19

This documentation is produced with the DocStrip utility, and required `\tau_N b \subset` with its `linegoal` option.

→ To get the package, run: `etex tabu.dtx`  
→ To get the documentation run (thrice): `pdflatex tabu.dtx`  
To get the index, run: `makeindex -s gind.ist tabu.idx`

The `.dtx` file is embedded into this pdf file thank to `embedfile` by H. Oberdiek.

1. Inside the `tabu` environment a few macros are modified... this was compulsory !

<b>6</b>	<b>Some other features</b> .....	<b>20</b>
6.1	Printing numbers inside <code>tabu</code> with <code>numprint</code> and <code>siunitx</code> .....	20
6.1.1	<code>\tabudecimal</code> .....	
6.1.2	<i>You should know how it works...</i> .....	
6.2	Paragraph indentation .....	21
6.3	<code>delarray</code> shortcuts .....	22
<b>7</b>	<b>Differences between <code>tabu</code>, <code>tabular</code>, <code>tabularx</code> and <code>longtable</code></b> .....	<b>22</b>
7.1	Paragraph indentation .....	22
7.2	Custom environments .....	22
7.3	Inversion of tokens .....	22
7.4	Improved process for rewriting columns ( <i>for keen readers</i> ) .....	23
<b>8</b>	<b>The package options</b> .....	<b>24</b>
8.1	The <code>debugshow</code> package option .....	24
8.2	The <code>delarray</code> package option .....	24
8.3	The <code>linegoal</code> package option .....	24
<b>9</b>	<b>Corrections of some bugs (<i>available only inside <code>tabu</code></i>)</b> .....	<b>24</b>
9.1	Correction for <code>colortbl</code> and <code>arydshln</code> : compatibility with <code>delarray</code> .....	24
9.2	Correction for <code>arydshln</code> : @ columns .....	25
<b>10</b>	<b>To do for even better <code>tabus</code></b> .....	<b>25</b>
<b>11</b>	<b>TECHNICAL NOTICE AND IMPLEMENTATION</b> .....	<b>26</b>
11.1	Drawing a tabular - The $\mathcal{T}_{\mathcal{N}bc}$ approach .....	26
11.2	Algorithms .....	26
11.3	The <code>tabu</code> strategies .....	30
11.4	Identification and Requirements .....	31
11.5	Flow chart of expansion .....	31
11.6	Some constants .....	33
11.7	Rules, colors and vertical adjustment .....	38
11.8	The entry inside <code>tabu</code> .....	47
11.9	The rewriting process: inside the “ <code>\@mkpream</code> group” .....	51
11.10	Implementing the strategy at the exit of the <code>\@mkpream</code> group .....	59
11.11	One trial after the other ( <code>\tabu@strategy</code> ) .....	61
11.12	The algorithms: Measuring the <code>tabu</code> box .....	64
11.13	Measuring the natural width of columns ( <code>varwidth</code> code from D. Arseneau) .....	69
11.14	Measuring the height and depths of rows .....	70
11.15	<code>\tabuphantomline</code> .....	71
11.16	Horizontal lines inside <code>tabu</code> : <code>\tabucline</code> , <code>\firsthline</code> and <code>\lasthline</code> .....	72
11.17	Numbers in <code>tabu</code> .....	75
11.18	Verbatim inside <code>tabu</code> with X columns .....	76
11.19	<code>\savetabu</code> .....	77
11.20	<code>\rowfont</code> .....	79
11.21	Taking care of footnotes and <code>\arraybackslash</code> .....	83
11.22	Corrections .....	85
11.23	Package options and Initialisation .....	86
<b>12</b>	<b>References</b> .....	<b>90</b>
<b>13</b>	<b>History</b> .....	<b>90</b>
	[2011/02/19 v2.5] .....	90
	[2011/02/17 v2.4] .....	90
	[2011/02/13 v2.3] .....	90
	[2011/02/12 v2.2 – <b>New implementation - Absolutely no modification of <code>array.sty</code></b> ] .....	90
	[2011/01/19 v2.1] .....	91
	[2011/01/18 v2.0] .....	91
	[2011/01/15 v1.9] .....	91
	[2010/12/28 v1.8] .....	91
	[2010/12/18 v1.7] .....	92
	[2010/12/07 v1.5] .....	92
	[2010/11/22 v1.4] .....	92
	[2010/11/18 v1.3] .....	92
	[2010/11/15 v1.2] .....	93
	[2010/10/28 v1.1] .....	93
<b>14</b>	<b>Index</b> .....	<b>93</b>

## Summary of the features provided by $\tau_{\text{N}}bc$

<code>tabu</code>	is like <code>tabular</code> in text mode and like <code>array</code> in math mode when there is no X column in its preamble
<code>longtabu</code>	is like <code>longtable</code> with the possibility to use <code>tabu</code> X columns and vertical lines with the extended syntax.
<code>{tabu} to &lt;dimen&gt;</code>	specifies the target width of the whole tabular. This is like <code>tabular*</code> with an automatic stretchability that can be overwritten with <code>@{\extracolsep {dimen}}</code> in front of the preamble.
<code>{tabu} spread &lt;dimen&gt;</code>	has no equivalent in L <sup>A</sup> T <sub>E</sub> X: the final width is <code>&lt;dimen&gt;</code> wider than the natural width that can be obtained with <code>spread Opt</code> .
<code>  [width,color]</code>	vertical lines have an optional parameter.
<code>X[coef,align,type]</code>	X columns widths are adjusted in order for the whole tabular to fit the target width. The target width is a dimension either:
<code>X[coef,align,type,\$]</code>	<ul style="list-style-type: none"> <li>→ directly specified with <code>{tabu} to&lt;dimen&gt;</code></li> <li>→ computed from the natural width: <code>{tabu} spread&lt;dimen&gt;</code></li> <li>→ by default <code>\linewidth</code> (or <code>\linegoal</code> with the <code>linegoal</code> package option).</li> </ul>
	<code>coef</code> scales the widths of the X columns, if there are more than one X column.
	<code>align</code> is either r, c, l or j (or <code>R C L J</code> ) and <code>type</code> can be p (default), m or b.
	<code>X[\$]</code> makes a math X column ( <i>ie.</i> <code>&gt;{\${}X&lt;{\${}</code> )
	<code>X[\$\$]</code> display math X column: <code>&gt;{\${\displaystyle }X&lt;{\${}</code>
<code>X[-coef,align,type]</code>	X columns widths are first computed with the absolute value: <code> coef </code> . Then the width is made narrower down to the natural width of the column if possible.
	In any case, the final width does not exceed the one obtained with <code>X[ coef ]</code> .
<code>X[X options]{S[S options]}</code>	Embed a <code>siunitx</code> S column into a <code>tabu</code> -X column.
<code>\everyrow {code}</code>	Allows to add horizontal lines automatically for every row. The settings can be changed inside the <code>tabu</code>
<code>\rowfont [align]{font spec}</code>	Modify the font and optionally the alignment of each cell in one row.
<code>\tabulinesep =&lt;dimen&gt;</code>	More control on vertical spacing of lines in a way very close to <code>cellspace</code> 's method (dynamic vertical spacing adjustment).
<code>\extrarowsep =&lt;dimen&gt;</code>	Control vertical spacing ( <code>\extrarowheight</code> and <code>\extrarowdepth</code> ): fixed vertical spacing adjustment. <code>\tabulinesep</code> generally gives better results.
<code>\tabudecimal {\usermacro }</code>	a help to align numbers easily inside a column.
<code>\savetabu {user-name}</code>	Saves the <code>tabu</code> preamble and its parameters. The command must appear at the end of a line.
<code>\usetabu {user-name}</code>	Makes a <code>tabu</code> of exactly the same shape as the one saved with <code>\savetabu</code> . All parameters ( <code>target</code> , <code>preamble</code> , <code>stretch etc.</code> ) are restored.
	This command is put alone in the preamble in place of the columns specifications.
<code>\preamble {user-name}</code>	Makes a <code>tabu</code> with the same preamble as the one saved with <code>\savetabu</code> . The only <code>preamble</code> is restored, not the <code>target</code> nor any other parameter.
	This command is put alone in the preamble in place of the columns specifications.

## Summary of the features provided by T<sub>nb</sub>

<code>\tabulinestyle {line spec}</code>	Sets the current line style to be used for <code> </code> and <code>\tabucline</code>
<code>\newtabulinestyle {name=spec,...}</code>	Defines a line style for use with <code>\tabucline [name]</code> or with <code>  [name]</code>
<code>\tabucline [spec]{start-stop}</code>	Draws a line comparable to <code>\hline</code> . The line <i>&lt;spec&gt;</i> can contain information for making a dash or dotted line (f.ex. <code>[on 3pt off 6pt]</code> ) and a color name. The line spec can also be defined with <code>\newtabulinestyle</code>
<code>\taburulecolor  dbl rule sep {rule color}</code>	sets the color for rules ( <code>\hline</code> , <code>\firsthline...</code> )
<code>\taburowcolors [skip]&lt;number&gt;{first .. last}</code>	Sets the color series to make alternate background colors for rows
<code>\tabuphantomline</code>	inserts a phantom ( <i>ie.invisible</i> ) line inside the <code>tabu</code> May be usefull with <code>\multicolumn</code> in some cases.
<code>\tracingtabu = 0, 1, 2, 3, 4</code>	Reports informations in the <code>.log</code> file about the steps of the algorithm for <code>tabu X</code> columns, and the informations saved by <code>\savetabu</code> .

# 1 Examples and counterexamples

Let's begin in colors !

$\tau_{\text{N}}b\subset$  provides facilities to put horizontal and vertical leaders in a tabular. The package `xcolor` must be loaded of course. Background colors for cells are left to package `colortbl` which is fully compatible with  $\tau_{\text{N}}b\subset$ .

## 1.1 “Locally global” settings and their scopes

`\tabulinestyle`  
`\taburulecolor`  
`\taburowcolors`  
`\everyrow`

$\tau_{\text{N}}b\subset$  observes T<sub>E</sub>X grouping levels for the settings of rule colors (`\taburulecolor`) and styles (`\tabulinestyle`), and `\everyrow`. There is however a subtlety for nested `tabu` environments as described in this example:

Listing 1: Locally global settings and their scopes

```
\taburulecolor |gray!50|{\red} \arrayrulewidth=1pt
{
  \taburulecolor |yellow|{\blue}
  \begin{tabu}{|X|X|} \hline
    Here the lines & are drawn in blue \\ \taburulecolor{\green} \hline
    But starting from here & they are green coloured ! \\ \hline
    And now a nested tabu & \begin{tabu}{X} \firsthline\hline
                                guess what colour \\ \hline
                                is used for rules ?\\ \lasthline\hline
                                \end{tabu} \\ \hline
  \end{tabu}
  % Inside the group, rule colors are blue
}
% After the group, rule colors are red again !
\begin{tabu}{X}\hline\hline\indent\end{tabu}
```

Color of the T <sub>E</sub> X group	Here the lines	are drawn in blue
	But starting from here	they are green coloured !
Color of the last end-of-line setting	And now a nested tabu	guess what colour
Color of the T <sub>E</sub> X group		is used for rules ?
Inside the group, rule colors are blue    After the group, rule colors are red again !		

The “rules” are the following:

- If outside of a `tabu` environment, the settings are local to the T<sub>E</sub>X group. Every tabular drawn inside this group will inherit from the settings of that group.
- If `\taburulecolor` (or `\everyrow` or `\tabulinestyle`) is used inside a cell of the tabular, this is the same: the settings a local to that cell, and any nested tabular will inherit from the setting of that cell.
- When used after the end of a row, the settings are globally changed from that point until the end of the tabular, or until a new setting is set at the end of a further row (T<sub>E</sub>Xnically, this is done inside a `\noalign` group). But a nested `tabu` does not inherit from this “global” setting, and inherits from the settings of the T<sub>E</sub>X group instead.

If `\arrayrulecolor` or `\doublerulesepcolor` (from package `colortbl`) are used instead of `\taburulecolor` then colors are globally overwritten.

A counterexample from the `xcolor` package: `\rowcolors` does not like `\cline`, `\cmidrule` etc.<sup>2</sup>

```
\rowcolors{2}{green!25}{yellow!50}
\begin{tabular}{cc} \toprule
\repeatcell{2{
  rows=5,
  text/col1=test ,
  text/col2=row \number\rownum} \\\
  test & other row \number\rownum \\\ \cmidrule
    {1-2}
  test & other row \number\rownum \\\
  test & other row \number\rownum \\\ \\\
    bottomrule
\end{tabular}
```

test	row 1
test	row 2
test	row 3
test	row 4
test	row 5
test	other row 6
test	other row 8
test	other row 9

The `\rownum` counter is not reliable in the case of `\cline` or `\cmidrule`.

In addition, the first coloured row is yellow, while one could have expected it green...

For  $\tau_{\text{N}}b\text{c}$  color changes are called at `\everyrow`:

```
\taburowcolors [2] 2{green!25 .. yellow!50}
\begin{tabu}{*2{X[c]}} \toprule
\repeatcell{2{
  rows=5,
  text/col1=test ,
  text/col2=row \thetaburow} \\\
  test & other row \thetaburow \\\ \cmidrule
    {1-2}
  test & other row \thetaburow \\\
  test & other row \thetaburow \\\ \\\
    bottomrule
\end{tabu}
```

test	row 1
test	row 2
test	row 3
test	row 4
test	row 5
test	other row 6
test	other row 7
test	other row 8

$\tau_{\text{N}}b\text{c}$  does not use “real” alternate colors but colorseries provided by package `xcolor`. This allow some gradations:

```
\taburowcolors 5{SkyBlue!65 .. Gold!60}
\begin{tabu}{X[-1]X}
\repeatcell 2{
  rows=10,
  text/col1=test ,
  text/col2={Row number
              \row$=\thetaburow},
}
\end{tabu}
```

test	Row number 1=1
test	Row number 2=2
test	Row number 3=3
test	Row number 4=4
test	Row number 5=5
test	Row number 6=6
test	Row number 7=7
test	Row number 8=8
test	Row number 9=9
test	Row number 10=10

## 1.2 X column widths computation

The new algorithm implemented in version 2.5 requires only one measure of the width of the table in any case. This speeds up the convergence of the algorithm.

```
\begin{tabu} to 140mm {X[1,1] | X[2,c] | X[3,c] | X[1,r]||}
|\dotfill | & Text & & Text & & Text \\\
Text & & Text & & Text & \\
\end{tabu}
```

Text	Text	Text	Text
1X= 17.5mm	2X = 35mm	3X = 52.5mm	1X = 17.5mm

$$X = (140mm - 8 \times \text{tabcolsep} - 5 \times \text{arrayrulewidth}) / 7 = 17.4896mm$$

2. Because color changes are done at `\everycr`, which is not exactly the same as  $\tau_{\text{N}}b\text{c}$  `\everyrow`!

## 1.3 Inserting Verbatim material (fancyvrb)

Though the content of the `tabu` environment is collected for measuring purpose, it is possible to insert verbatim material with the `tabu*` variant of the environment. The content is then carefully collected and re-scanned (with `\scantokens`). During the process, the `@` letter is read with the category code it has been given at the entry inside the environment (it is possible to say `\makeatletter` before `\begin{tabu*}`).

Example:

It is possible to insert Verbatim material with some `\csname` control sequences `\endcsname` inside a `tabu` and inside `X` columns. Negative coefficients work well too, adjusting the width of the `X` column to the natural width if it is finally less than the width computed with the absolute value of the coefficient.

A complete `Verbatim` environment is also admissible.

But you must use the star form of the environment: `tabu*` which uses `\scantokens`.

Verbatim environments must be put alone on their lines (in the input file) for nothing is allowed after `\begin{Verbatim}` or `\end{Verbatim}`.

Another point to know is that `\begin` and `\end` control sequences should match otherwise, you must enclose the `Verbatim` environment inside braces.

This is related to the fact that `tabu` collects its body, and looks for matching pairs of `\begin ... \end` !

`tabu*` is useless when nested inside another tabular. The star form of the environment should be used only for the outermost table ! Comments are removed, unless the `%` character is given a category code of 12 (or 11) before the entry inside the environment.

```
\tabulinestyle{on2pt Crimson!60 off3pt yellow!50} \tabulinesep=1mm
\makeatletter \@makeother\%
\begin{tabu*}spread 0pt {|X[-1]X|} \tabucline -
This is a small \Verb+\Verbatim+\par
insertion
&
\begin{Verbatim}[listparameters={\topsep=-\ht\strutbox}]
And this is a complete % with some comments
Verbatim environment % every now and then
\end{Verbatim}
\\ \tabucline -
\end{tabu*}
```

Here a small <code>\Verbatim</code> insertion	And this is a complete <code>%</code> with some comments Verbatim environment <code>%</code> every now and then
---	--

It's not possible to insert a `lstlisting` environment presently, but you can save such an environment in a `\vbox` and insert it inside the `tabu` of course.

## 1.4 Maths inside tabu X columns

```
$\begin{tabu}spread .5in |{*3{X[$c]}}|
\alpha & \beta & \gamma \\
\sum_i \frac{a_i}{x_i} & 0 & \cdot \\
\end{tabu}$
```

X[\$] columns

$$\left| \begin{array}{ccc} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \end{array} \right|$$

```
$\begin{tabu}spread .5in |{*3{X[$$c]}}|
\alpha & \beta & \gamma \\
\sum_i \frac{a_i}{x_i} & 0 & \cdot \\
\end{tabu}$
```

X[\$\$] columns

$$\left| \begin{array}{ccc} \alpha & \beta & \gamma \\ \sum_i \frac{a_i}{x_i} & 0 & \cdot \end{array} \right|$$







ragged2e settings) and the **column type** (p, m, or b).  
 tabu has a default target width when used with X columns, making nesting even easier.

- You are used to the tabular environment in text mode, and array environment in math mode, but tabu works in both modes and its name does not change... X columns are also possible in math mode; delarray shortcuts for delimiters are available in both math and text modes.
- A tabu environment can contain another tabular of any kind: tabular, tabular\* , tabularx or tabu itself can be placed in any cell of a tabu. Conversely, tabu can be placed in a tabular, tabularx etc..
- tabu provides facilities for vertical and horizontal lines, and for the insertion of verbatim text inside X columns.
- tabu is fully compatible with colortbl, delarray, hline, makecell, booktabs, siunitx, dcolumn, warpcol, etc.. When you are inside a tabu environment, you can use \raggedleft, \raggedright and \centering without special care about \arraybackslash and conversely \\ has its “normal” meaning inside a list of items that may appear in a X column...

\begin {tabu} to<dimen> is like tabular\* but the inter-columns space is given a stretchability of 1fil, in other words @{\extracolsep {0pt plus 1fil}} is inserted by default at the beginning of the tabular preamble, unless another value for \extracolsep is specified. Therefore “tabu to” fills in width the specified <dimen>.

\begin {tabu} spread<dimen> does a tabular whose width is <dimen> wider than its natural width. @{\extracolsep {0pt plus 1fil}} is inserted by default if <dimen> > 0.

## 2.2 longtabu, longtabu to and longtabu spread

```
\begin {longtabu} [l | c | r] {tabular preamble}
\begin {longtabu} to <dimen> [l | c | r] {tabular preamble}
\begin {longtabu} spread <dimen> [l | c | r] {tabular preamble}
```

longtabu is just like tabu but page breaks are allowed between rows of the table. longtabu is based on the longtable package which must be loaded, and all features of the longtable environment works inside longtabu: \endhead, \endfirsthead, \endfoot, \endlastfoot and \caption.

longtabu enhances the longtable environment with the possibility to use X columns and line specifications for horizontal and vertical rules. longtabu is thus much easier than ltable.

The following commands provided for tabu do not work with longtabu:

tabu command	Not available	Not implemented	Comment
\tabucline		✖	\tabucline does not care of page breaks presently: use \hline instead.
\usetabu	✖		but \savetabu and \preamble work.
mathematical mode	✖		longtable is not designed to work in math mode.
delarray shortcuts	✖		a delimiter cannot be spanned over pages...
\tabuphantomline	✖		useless inside longtabu

However, tabu X columns, \rowfont, \extrarowsep, \tabulinesep, \tabudecimal work inside longtabu.



tabularx is reimplemented and enhanced for tabu X columns. `\tabuDisableCommands` can be used to neutralize the expansion of additional macros during the trials.

## X columns with “tabu spread”

`tabu X` columns can be used with “`tabu spread`” to adjust the column widths of tabulars that contain only small pieces of text. The question is: how to make a tabular the width of the line, with 6 columns; the columns 1, 2, 5 and 6 are of equal widths and the widths of columns 3 and 4 are only one half. As possible solution:

```
\begin{tabu} to\linewidth{|X[2]|X[2]|X|X|X[2]|X[2]|}\hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

But the text in each cell is very short: one single character, and you prefer the table to be tight, but don't know the exact width of the whole:

```
\begin{tabu} spread 0pt{ |X[2]|X[2]|X|X|X[2]|X[2]| } \hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

But now it's definitely too narrow, then give it some more space:

```
\begin{tabu} spread 2in{ |X[2]|X[2]|X|X|X[2]|X[2]| } \hline
1 & 2 & 3 & 4 & 5 & 6 \\ \hline
\end{tabu}
```

1	2	3	4	5	6
---	---	---	---	---	---

`tabu spread` is useless with long columns: the following tabular was made with this preamble:

$$\begin{array}{c} \text{spread } 3\text{cm}\{\textcircled{\scriptsize X}[9]\textcircled{\scriptsize X}[4] \mid \textcircled{\scriptsize X} \mid \} \end{array}$$

"Like the air we breathe, Sherlock Holmes is everywhere. His pipe-smoking, deer stalkered image peers at us from ads in Yellow Pages, to signs for neighbourhood crime-watch; from billboards to the classroom; from film and television to the public library, and now over the Internet. He long ago transcended the boundaries of 19th Century London<sup>3</sup> to become an international best-seller and has been accepted as part of British folklore. Holmes is alive to millions."

There the text was too long,  
and `tabu spread` behaves as if  
you didn't give it a target.

The result of this example is the same as if one had written `\begin{tabu}to\linewidth`.

**Sherlock Holmes**

---

The “official” web site: <http://www.sherlockholmes.com/>

In the preamble, @{} means that the margin is removed.

### Negative width coefficients for X columns

```
\tabulinesize{3pt ForestGreen}
\begin{tabu}{|X[-1m]|X[c m]|}
    \tabucline - \savetabu{FirstNegativTest}
    $\begin{tabu}({X[-1$]X[-1$c]})
        \alpha & \beta & \\
        \gamma & \delta + \epsilon + \zeta + \eta + \theta \\
    \end{tabu}$
    &
    This is a tabu with negativ width coefficients for \texttt{X} columns
    \\ \tabucline -
\end{tabu}
```

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta + \epsilon + \zeta + \eta + \theta \end{pmatrix} \quad \text{This is a tabu with negativ width coefficients for } \mathbf{X} \text{ columns}$$

---

3. Capital of the U.K. (too see a linked footnote)

$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta + \epsilon + \zeta + \eta + \theta \end{pmatrix}$	And this is the same with <code>\tabulinesep</code> set to 2pt.
--	---

## Multicolumn in tabu

### `\tabuphantomline`

The process of `\multicolumn` implies the T<sub>E</sub>X primitive `\omit` which discards the tabular preamble for the spanned columns. Discarding the preamble means discarding the information about the widths of the columns. This explains why the following example does not work properly:

```
\begin{tabu}{|X|X|X[2]|} \tabucline-
\multicolumn2{|c|}{Hello} & World \\ \tabucline-
\end{tabu}
```

The correct result can be obtained by the mean of a phantom line, that will remain invisible unless your preamble contains special `@` or `!` columns that prints some text:

```
\begin{tabu}{|X|X|X[2]|} \tabucline-
\multicolumn2{|c|}{Hello} & World \\ \tabucline-
\tabuphantomline
\end{tabu}
```

Hello	World
-------	-------

Remember you may need `\tabuphantomline` in conjunction with `\savetabu` and `\usetabu` with `\multicolumn`. Even if it is possible to add a `\tabuphantomline` in any line of the `tabu`, it is a good practice to append it *at the end* of the `tabu`, for it may introduce undesirable side effects on vertical alignment otherwise, when `tabu` is nested inside another tabular.

In particular, `\tabuphantomline` should not be followed by `\cr` or `\\` or `\tabularnewline`...

The need for this command could disappear in a future release, but this requires a complete new implementation of `\multicolumn`...

## 2.4 `\tabulinesep` and `\extrarowsep` – Mastering vertical space

```
\tabulinesep = <dimen>
\tabulinesep = ^<dimen>
\tabulinesep = _<dimen>
\tabulinesep = ^<dimen>_<dimen>
\tabulinesep = _<dimen>^<dimen>
```

`\tabulinesep` sets the *minimal* vertical space allowed between the cell content and the cell border. The macro may be prefixed by `\global` (even inside a `\noalign` group)<sup>4</sup>.

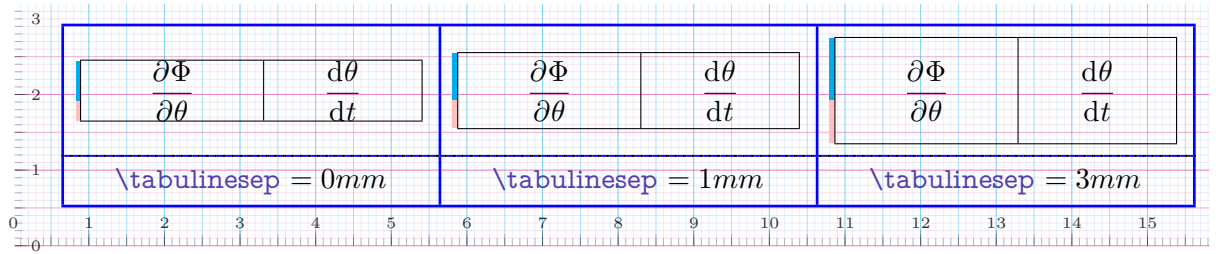
It is possible to set the “top limit” (a T<sub>E</sub>X dimension called `\abovetabulinesep`) and the “bottom limit” independently with the syntaxes:

```
\tabulinesep = ^<dimen>      sets \abovetabulinesep
\tabulinesep = _<dimen>      sets \belowtabulinesep
\tabulinesep = _<dimen>^<dimen> sets \belowtabulinesep and \abovetabulinesep.
```

These parameters can be used in text and math modes to give more vertical space between lines, especially when using math formulae.

Examples (with `\tracingtabu = 3` and `interfaces-\papergraduate` to see the struts):

4. However `\tabulinesep` is not a dimension ! You can’t test, for example, `\ifdim \tabulinesep > 0pt` ! Test `\abovetabulinesep` and `\belowtabulinesep` instead, if needed.



$\backslash\text{tabulinesep}$  is a soft parameter, and leads to rows which do not share the same height.

$\backslash\text{extrarowsep} = \langle \text{dimen} \rangle$
$\backslash\text{extrarowsep} = ^\langle \text{dimen} \rangle$
$\backslash\text{extrarowsep} = _\langle \text{dimen} \rangle$
$\backslash\text{extrarowsep} = ^\langle \text{dimen} \rangle _\langle \text{dimen} \rangle$
$\backslash\text{extrarowsep} = _\langle \text{dimen} \rangle ^\langle \text{dimen} \rangle$

$\backslash\text{extrarowsep}$  is an extra vertical space which is added to each row, inconditionally. `array.sty` provides the T<sub>E</sub>X dimension  $\backslash\text{extrarowheight}$  and  $\tau_{\text{Nbc}}$  provides  $\backslash\text{extrarowdepth}$  in addition. As a result, the rows can share the same height/depth but the spacing is not dynamic.  $\backslash\text{tabulinesep}$  can be used even with positive values for  $\backslash\text{extrarowsep}$ , for `tabu` inserts only one strut per row and vertical spacing computations are possible in all cases.

The macro can be prefixed by  $\backslash\text{global}$  as well, even inside a  $\backslash\text{noalign}$  group<sup>5</sup>.

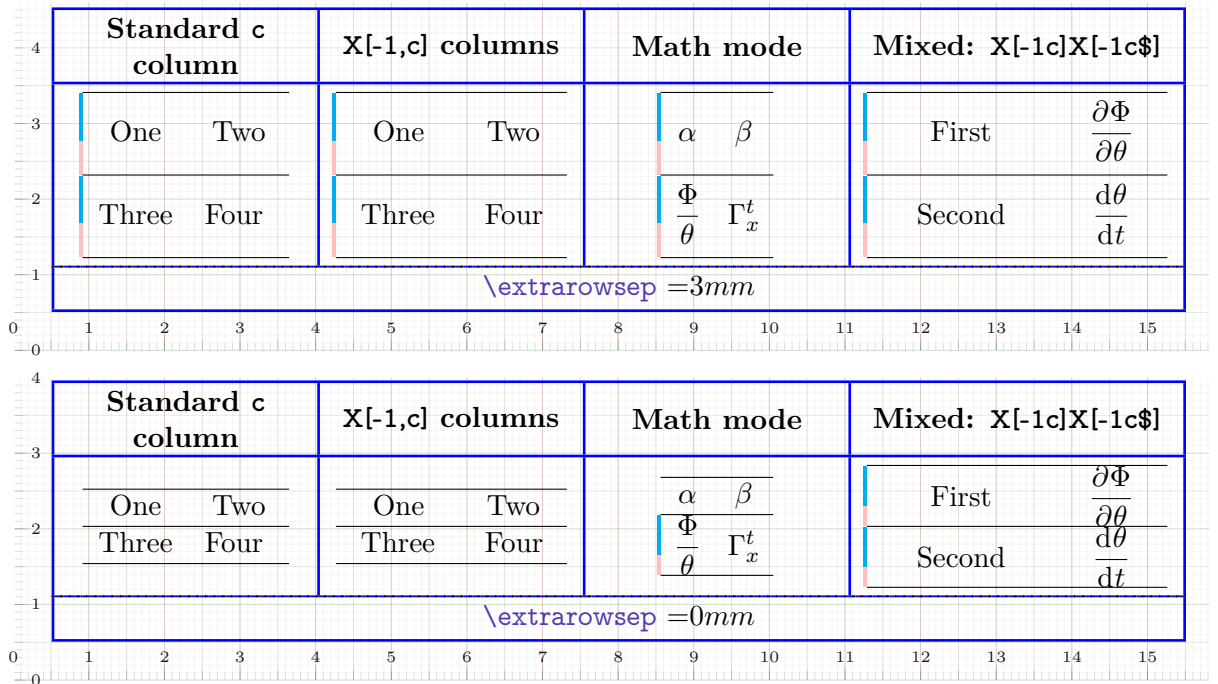
Set  $\backslash\text{extrarowheight}$  and  $\backslash\text{extrarowdepth}$  to different values, with the syntaxes:

$\backslash\text{extrarowsep} = ^\langle \text{dimen} \rangle$	sets $\backslash\text{extrarowheight}$ $\backslash\text{extrarowdepth}$ is unchanged
$\backslash\text{extrarowsep} = _\langle \text{dimen} \rangle$	sets $\backslash\text{extrarowdepth}$ $\backslash\text{extrarowheight}$ is unchanged
$\backslash\text{extrarowsep} = _\langle \text{dimen} \rangle ^\langle \text{dimen} \rangle$	sets $\backslash\text{extrarowdepth}$ and $\backslash\text{extrarowheight}$ .

Both  $\backslash\text{extrarowheight}$  and  $\backslash\text{extrarowdepth}$  are scaled by  $\backslash\text{arraystretch}$  (a scaling *macro*<sup>6</sup> of `array.sty`) if  $\backslash\text{arraystretch} > 1$ ...

These parameters can be used in text and math modes.

Examples (with  $\backslash\text{tracingtabu} = 3$  and `interfaces-\papergraduate` to see the struts):



5. However  $\backslash\text{extrarowsep}$  is not a dimension ! You can't test, for example,  $\backslash\text{ifdim} \backslash\text{extrarowsep} > 0pt$  ! Test  $\backslash\text{extrarowheight}$  and  $\backslash\text{extrarowdepth}$  instead, if needed.  
6.  $\backslash\text{arraystretch}$  is not a dimension but a macro that stores a scaling factor.

## 2.5 tabu in math mode

On the left, you can see the famous Maxwell-Lorentz equations for electromagnetic field in vacuum, published in 1873.

$$\left\{ \begin{array}{l} \operatorname{div} \vec{E} = \frac{\rho}{\epsilon_0} \\ \operatorname{div} \vec{B} = 0 \\ \operatorname{rot} \vec{E} = -\frac{\partial \vec{B}}{\partial t} \\ \operatorname{rot} \vec{B} = \mu_0 \vec{j} + \mu_0 \epsilon_0 \frac{\partial \vec{E}}{\partial t} \end{array} \right.$$

In this example, the big `tabu` is: `\begin{tabu} to\linewidth {XX[-1$]}`.

The nested `tabu` (in math mode) uses `delarray` shortcut: its preamble is: `\begin{tabu}(\rl)`.

`\tabulinesep` has been set to `3pt`.

Horizontal rules are `booktabs` `\toprule` and `\bottomrule`.

array	tabu	tabu spread 1em
$\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$	$\begin{tabu}(\rl)\alpha \beta \\ \gamma \delta \end{tabu}$	$\begin{tabu}(\rl)\alpha \beta \\ \gamma \delta \end{tabu}$

Here, vertical lines are made with `delarray` shortcuts: `\begin{tabu} spread 1em |{cc}|`

Vertical lines inside the tabular preamble gives:

$$\begin{array}{cc} \alpha & \beta \\ \gamma & \delta \end{array}$$

This was an example of `\savetabu ... \usetabu` to keep the alignment.

## 3 Lines leaders and colors inside tabu

### 3.1 First important remark

The features provided in this section are quite experimental: they are not generally taken for good typography. You can use `tabu` with package `booktabs` for example, which provides properly designed commands for horizontal rules in tabulars. `arydshln` is pretty good too, but it modifies a huge amount of macros of `array.sty`, something that  $\tau_{\text{N}bc}$  does not.

Lines in `tabu` printed in this document are mostly made with `booktabs`.

### 3.2 Vertical lines: | has an optional parameter

Inside `tabu` environment, the vertical line marker `|` has an *optional* argument which is the width of the vertical rule. The default width remains `\arrayrulewidth` of course. The optional argument for `|` can also contain the name of a color. *color names* are only possible, not a color specification by the mean of a color model. The width of the line if specified, must come before the color name and... as for `X` columns parameters, commas are optional.

Example:

$\begin{array}{ c c } \hline \text{Hello} & \text{World} \\ \hline \end{array}$	The <code>tabu</code> you see on the left was made with the code on the right.	<pre>\begin{tabu}{  [5pt] c c [5pt] } Hello &amp; World \end{tabu}</pre>
$\begin{array}{ c c } \hline \text{Hello} & \text{World} \\ \hline \end{array}$	The <code>tabu</code> you see on the left was made with the code on the right.	<pre>\begin{tabu}{  [5pt red] c c [5pt Indigo] } Hello &amp; World \end{tabu}</pre>

This example was printed inside a `tabu*` whose preamble is: `X[-1m] X[m] X[-2m]`



It is not a necessary to protect the optional argument with braces: `[{...}]`. because  $\tau_{\text{N}}b\text{c}$  takes care the `|` token to be rewritten before any other column type (the same for `tabu X` columns, and `siunitx S` columns). The rewriting process is divided into three stages under control inside a `tabu` environment.

### 3.3 Multiple `\firstline` and `\lastline`

<code>\firstline</code> [extratabsurround]	make multiple lines !
<code>\firstline</code> [extratabsurround]\hline	
<code>\lastline</code> [extratabsurround]	
<code>\lastline</code> [extratabsurround]\hline	

`\firstline` and `\lastline` are defined in `array.sty` and can be used to preserve the alignment of text, when using horizontal lines. Besides, the optional argument can be used to change (locally) the `\extratabsurround` dimension.

The example of `array` documentation is:

<div> <div>Tables</div> <div>with no line commands used</div> <div>versus</div> <div>tables</div> <div>with some line commands</div> <div>used.</div> </div>	<div> <div>Tables</div> <div>with no line commands used</div> <div>versus</div> <div>tables</div> <div>with some line commands</div> <div>used.</div> </div>
with <code>\firstline</code> and <code>\lastline</code>	with <code>\hline</code> (text alignment is not preserved)

Now with `tabu` you can make double, triple (or more) `\firstline` or `\lastline` as in:

Top alignment	<div> <div>Tables</div> <div><code>\begin{tabu}[t]{c}</code></div> <div>with no<code>\line</code> <code>\commands</code> <code>\used</code></div> <div><code>\end{tabu}</code></div> <div>versus tables</div> <div><code>\begin{tabu}[t]{ c }</code></div> <div><code>\firstline \hline \hline \hline</code></div> <div>with some <code>\line</code> <code>\commands</code> <code>\</code></div> <div><code>\lastline \hline \hline \hline</code></div> <div><code>\end{tabu}</code> used.</div> </div>	<div> <div>Tables</div> <div>with no line commands used</div> <div>versus</div> <div>tables</div> <div>with some line commands</div> <div>used.</div> </div>
Bottom alignment	<div> <div>Tables</div> <div><code>\begin{tabu}[t]{c}</code></div> <div>with no<code>\line</code> <code>\commands</code> <code>\used</code></div> <div><code>\end{tabu}</code></div> <div>versus tables</div> <div><code>\begin{tabu}[b]{ c }</code></div> <div><code>\firstline \hline \hline \hline</code></div> <div>with some <code>\line</code> <code>\commands</code> <code>\</code></div> <div><code>\lastline \hline \hline \hline</code></div> <div><code>\end{tabu}</code> used.</div> </div>	<div> <div>Tables</div> <div>with no line commands used</div> <div>versus</div> <div>tables</div> <div>with some line commands</div> <div>used.</div> </div>

`\firstline \firstline \firstline` is equivalent to: `\firstline \hline \hline`  
and also to: `\firstline \hline \hline \hline`

But the optional argument must come in *first position*: `\firstline` [extratabsurround] ...

The same for `\lastline`.

In yellow you can see the `\extratabsurround` strut, because `\tracingtabu = 3` for this `tabu`



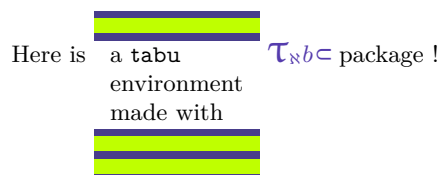
### 3.4 More style for lines

`\taburulecolor`  $\{\langle rule\ color\rangle\}$   
`\taburulecolor`  $|\langle double\ rule\ sep\ color\rangle| \{\langle rule\ color\rangle\}$

`\taburulecolor` sets (in a “locally-global” way) the color to be used for `\hline`, `\firsthline`, `\lasthline` and also vertical lines if the standard line style is used (the standard line style is active after `\tabulinestyle`  $\{\}$  or after `\tabureset`).

The optional parameter enclosed by vertical bars:  $|\langle double\ rule\ sep\ color\rangle|$  is the color to set between two adjacents rules. If not specified, double (or triple...) rules are separated by a vertical space (`\vskip`).

```
\taburulecolor |lime|{DarkSlateBlue}
\arrayrulewidth=1mm \doublerulesep=2mm
Here is
\begin{tabu}spread 0pt {X[-1]}
\firsthline\hline
a tabu \\\
environment \\\
made with \\\ \lasthline [5mm]\hline\hline
\end{tabu} \TabU package !\par
And the next paragraph follows...
```



And the next paragraph follows...

`\tabulinestyle`  $\{\langle line\ style\ specification\rangle\}$

`\tabulinestyle` sets the line style for vertical (`|`) and horizontal lines (*ie.* `\tabucline`: `\hline`, `\firsthline` etc. are not modified by `\tabulinestyle`)

The line specification is of the form:

3pt rule color on 4pt dash color off 5pt gap color  
 rule color on 4pt dash color off 5pt gap color  
 on 4pt dash color off 6pt gap color  
 3pt rule color  
 on 4pt dash color  
 off 5pt  
 3pt  
 Named style defined by `\newtabulinestyle`

Well... any parameter is optional. Obviouly the rule color is the same as the dash color and the former overwrites the latter if both are given.

Your color names can contain spaces but:

- If the first character in the line specification is not a letter, then it is taken as a dimension: the thickness of the line. Otherwise, the default thikness is used *ie.* `\arrayrulewidth`.
- Your color names must not contain any series of characters that match one the patterns:  

on?
off?

where `?` is a character of category 12, different from `!` and possibly preceded by spaces. I don't think this is a real limitation...

`\newtabulinestyle`  $\{\langle style=line\ spec.,\ style=line\ spec.,\ ...\rangle\}^{\text{babel}}$

This command defines a line style to be used in the first optional argument of `\tabucline` (horizal lines) or the optional argument of `|` (vertical lines) or with `\tabulinestyle` (locally-global style).

Style names and color names are babel-protected.


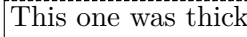
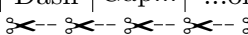


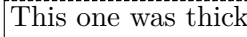
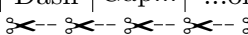


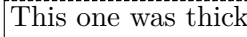
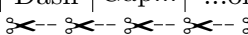

`\tabucline`  $[\text{style or spec.}]\{\text{start-end}\}$

`\tabucline` is an attempt to give a versatile command to make horizontal lines:

- `\tabucline` is pretty good with vertical lines even if the thickness of the line grows up,

- `\tabucline` takes care of `\extrarowheight`,
- `\tabucline` can make horizontal dashed lines, with a pgf/TikZ syntax:  
`\tabucline [ $\langle width \rangle$  on $\langle dash \rangle$  off $\langle gap \rangle$ ]{ $\langle first column \rangle$ - $\langle last column \rangle$ }`
- alternatively, you can give `\tabucline` a `\hbox` to make a leader with it: The  $\langle spec. \rangle$  must then begin with `\hbox`, `\box` or `\copy`,
- finally you can give `\tabucline` a color *name*, after the line specification.

Any parameter can be omitted.

	<code>\tabucline [1pt on 1.5pt off 2pt]{1-4}</code>	draws a horizontal dashed line of width 1pt. Dashes are 1.5pt long and gap width is 2pt. The line is drawn between columns 1 and 4. Here there are only 2 columns and the line stops at column 2.																		
[1pt on 1.5pt off 2pt]	<code>\tabucline [1.5pt]{-}</code>	draws a horizontal solid line of width 1.5pt between the first and the last column.																		
[1.5pt]	<code>\tabucline {2-}</code>	draws a horizontal solid line of width <code>\arrayrulewidth</code> between the second column and the last one.																		
default	<code>\tabucline [on 2pt red]{-5}</code>	draws a horizontal dashed line between columns 1 and 5 of width <code>\arrayrulewidth</code> . Dashed are 2pt long and gap width is 4pt (the default).																		
[on 2pt red]																				
*****																				
Define the line style		Use the line style																		
<code>\newtabulinestyle {myline=0.4pt on 2.5pt off 1pt red}</code>		<code>\tabucline [myline]{-}</code>																		
Or use a leader or a box to make a leader with it directly in the argument of <code>\tabucline</code>																				
<code>\tabucline [\hbox {\$\scriptstyle \star \$}]{1-3}</code>																				
*****																				
<table><tr><td>Dashed or dotted</td><td>Dash</td><td>Gap</td><td>Dash</td><td>Gap...</td><td>...or leader</td></tr><tr><td>And below is the default</td><td></td><td></td><td></td><td></td><td></td></tr><tr><td></td><td>This one was thick</td><td></td><td>Interesting ?</td><td></td><td></td></tr></table>			Dashed or dotted	Dash	Gap	Dash	Gap...	...or leader	And below is the default							This one was thick		Interesting ?		
Dashed or dotted	Dash	Gap	Dash	Gap...	...or leader															
And below is the default																				
	This one was thick		Interesting ?																	

### 3.5 Automatic horizontal lines and row colors

`\everyrow {code}`

`\everyrow` can be used to insert horizontal lines automatically:

```
\begin{tabu}to .5\linewidth{cX[2mc]X} \tabucline[1pt]-
\everyrow{\tabucline[on 2pt]-}
This is      &a small example &of a \texttt{tabu}      &\&\&
which        &automatically  &&inserts                &\&\&
a horizontal &line after    &each of its row \everyrow{} &\&\& \tabucline[1
pt]-
\end{tabu}
```

This is	a small example	of a tabu
which	automatically	inserts
a horizontal	line after	each of its
		row

`\everyrow` can be used in `longtabu` as well. The syntax is like `\everycr`: a token-like syntax, and braces are mandatory: `\everyrow {argument}`.

`\taburowcolors` [first line]⟨*number*⟩{first .. last}

`\taburowcolors` sets the alternate colors to be used on every row of the tabular. The command can be used before a `tabu` environment or inside it, at the end of a row.

The optional parameter [first line] tells the first row from which background colors are starting – this optional parameter has no effect when `\taburowcolors` is used at the end of a row: background are starting immediately in this case.

⟨*number*⟩ is the number of colors in the color series. If not specified, it defaults to 2 (for alternate rows color).

Finally ⟨*first*⟩ and ⟨*last*⟩ are the first and the last colors in the colorseries.

Example:

```
\taburowcolors [2] 3{Crimson!30 .. ForestGreen!40}
\taburulecolor |GreenYellow|{OrangeRed}
\arrayrulewidth=1pt \doublerulesep=1.5pt
\everyrow{\hline\hline}
\begin{tabu} {X[-1]X}
This is      &just a test          \\
and i think &it will                \\
look        &rather bad             \\
for         &i've not               \\
chosen      &the colors             \\
with care.  &i can't               \\
say         &less...               \\
\taburowcolors 2{Crimson .. ForestGreen}
1           &This is Crimson       \\
2           &This is ForestGreen   \\
3           &This is Crimson       \\
4           &This is ForestGreen   \\
\end{tabu}
```

This is	just a test
and i think	it will
look	rather bad
for	i've not
chosen	the colors
with care.	i can't
say	less...
1	This is Crimson
2	This is ForestGreen
3	This is Crimson
4	This is ForestGreen

`\tabureset`

To go back to “standard” parameters,  $\mathcal{T}_{\text{N}bc}$  provides the command `\tabureset` which basically does:

<code>\tabulinesep = 0pt</code>	<code>\extrarowsep = 0pt</code>	<code>\extratabsurround = 0pt</code>
<code>\tabulinestyle {}</code>	<code>\everyrow {}</code>	<code>\taburulecolor   {} </code>
<code>\taburowcolors {}</code>		

## 4 Modifying the font and the alignment in one row: `\rowfont`

`\rowfont` [alignment]{font specification}

Inside a `tabu` environment, you can modify the font for each cell in a row. `\rowfont` has priority over column font specification, exactly like `\rowcolor` (package `colortbl`) has priority over `\columncolor`.

The alignment of each cell in one row can also be changed to:

<code>l</code> = left	or for <code>ragged2e</code> settings:	<code>L</code>
<code>c</code> = center		<code>C</code>
<code>r</code> = right		<code>R</code>
<code>j</code> = justify		<code>J</code>

Any other value for the optional ⟨*alignment*⟩ parameter is silently ignored. If `ragged2e` is not loaded, `L` `R` `C` and `J` are synonymous with the lowercase equivalent.

```
\begin{tabu}{|X|X[-1]|} \tabucline-
\rowfont[c]\bfseries
This &Is \\
\package{tabu} &\package \\
\\ \tabucline[on 2pt,blue]-
\\ \tabucline[off 2pt blue]-
```

```
\rowfont[r]\itshape
for          &\textt{tabu} and \textt{longtabu} \\ \tabucline-
\end{tabu}
```

This	Is
tabu	package
	<i>for tabu and longtabu</i>

## 5 Saving and restoring a tabu

`\savetabu {⟨user-name⟩}`

The command `\savetabu` can be used at the end of any line of a `tabu` environment to save the parameters of a `tabu` environment. The saving is always global. This allows to easily make tabulars which share exactly the same shape throughout your document. This can also be used as a kind of `tabbing` environment which is able to remember the tabs positions...

If the `⟨user-name⟩` has been used before, an info is displayed in the `.log` file and the previous settings are overwritten.

With the `\tracingtabu > 0`, informations about the saved parameters are reported in the `.log` file.

Recalling saved parameters are done with `\usetabu` (complete recovery) or `\preamble` (partial recovery of the preamble only).

`\usetabu {⟨user-name⟩}`

`\usetabu` is the complement of `\savetabu`: it can be put alone in the `tabu` preamble instead of the usual columns specifications to restore any previous settings saved with `\savetabu`.

The `⟨user-name⟩` must exist otherwise, you get an error.

`\usetabu` is a help to **make several tabulars of exactly the same shape, same target, same preamble**. The only parameter that can be changed is the optional vertical position parameter for the whole tabular.

`\usetabu` does not work with `longtabu`.

`\usetabu` locally restores:

- the preamble<sup>7</sup>.
- the vertical position `[c]`, `[b]` or `[t]`, unless another position is specified.
- the target width of the `tabu` in points: the saved target width does not contain any control sequence: it is fixed and stored in points.
- the width of `tabu X` columns: those widths are not calculated any more – even in the case of negativ coefficients – and `X` columns are directly transformed into `p`, `m` or `b` columns of the same widths as the ones that where calculated at the time of `\savetabu`
- `\tabcolsep` (or `\arraycolsep` in math mode) `\extrarowheight`, `\extrarowdepth`, `\arraystretch` and `\extratabsurround`
- `\arrayrulewidth`, `\doublerulesep` and the parameters for `\everyrow` `\taburulecolor`, `\tabulinestyle`, and `\taburowcolors`
- `\minrowclearance`, (package `colortbl`)

`\abovetabulinesep` and `\belowtabulinesep` are not restored, because they are related to the content of the tabular rather than to its shape.

Example:

```
\tabcolsep=12pt \extrarowsep=1mm
\tabulinestyle{on 1pt ForestGreen}
```

7. The complete `\halign`-preamble is restored.

```
\begin{tabu}to .7\linewidth{[XXX|X[c]]} \savetabu{mytabu} \tabucline -
This & is & tabu & package \\ \tabucline -
\end{tabu}
```

This	is	tabu	package
------	----	------	---------

```
\tabureset
\begin{tabu}{\usetabu{mytabu}} \tabucline -
\multicolumn{3}[c]{This is tabu} & package \\ \tabucline -
\tabuphantomline
\end{tabu}
```

This is tabu			package
--------------	--	--	---------

If one day you use `tabu`, you will have the idea to restore a `tabu` while modifying its target, or adding new columns... `\savetabu` and `\usetabu` have not been thought for this purpose, and you may have unexpected results.

`\preamble {⟨user-name⟩}`

`\preamble` can also be used after `\savetabu`. This is a variant of `\usetabu` that locally restores:

- the `tabu` (or `longtabu`) preamble.
- the vertical position [c], [b] or [t] (or [c], [l] or [r] for `longtabu`), unless another position is specified.
- the `tabu` / `longtabu` target width, unless another target is specified.

Any other tabular parameter is not restored.

Put `\preamble {⟨user-name⟩}` alone inside the `tabu` (or `longtabu`) preamble in place of the usual columns specifications.

`\preamble` works exactly as if you defined a `custom environment` for `tabu`.

`\preamble` works with `longtabu` .

Example (continued...):

```
\tabulinestyle{1pt off1pt}
\begin{tabu} to\linewidth{\preamble{mytabu}} \tabucline -
This & is & tabu & package \\ \tabucline -
\end{tabu}
```

This	is	tabu	package
------	----	------	---------

`\tabcolsep`, rule colors etc. are not restored from `\savetabu`: the only `tabu` preamble is restored.

## 6 Some other features

### 6.1 Printing numbers inside tabu with numprint and siunitx

`\tabudecimal`

$\tau_{\text{N}}b\text{c}$  provides a *facility* to print numbers inside columns. This facility is not implemented to replace `siunitx` `S` and `s` columns or `numprint` `n` and `N` columns or other packages that provide alignment such as `warpcol`, `dcolumn` or `rccol`. It just make easy to apply a macro you get already on each number in a column of a `tabu`.

`\tabudecimal` has been developped mainly because it makes possible to align numbers inside `tabu` `X` columns.



## 6.3 delarray shortcuts

When you enclose your tabular with math delimiters using `delarray` shortcuts,  $\tau_{\text{N}}b\subset$  tries to reach its target for the whole: the tabular and the delimiter(s). You can see the difference:

with overflow hboxes  
(17.5pt and 17.8pt two wide)  
for tabularx

$\left( \begin{array}{c} \text{This is } \texttt{tabu} \text{ with delar-} \\ \text{ray shortcuts for paren-} \\ \text{thesis around.} \end{array} \right)$	$\left\{ \begin{array}{c} \text{This is } \texttt{tabu} \text{ with de-} \\ \text{larray shortcuts for curly} \\ \text{brackets around.} \end{array} \right\}$
$\left( \begin{array}{c} \text{This is } \texttt{tabularx} \text{ with de-} \\ \text{larray shortcuts for paren-} \\ \text{thesis around.} \end{array} \right)$	$\left\{ \begin{array}{c} \text{This is } \texttt{tabularx} \text{ with de-} \\ \text{larray shortcuts for curly} \\ \text{brackets around.} \end{array} \right\}$

Here `\tabulinesep = 3mm`

## 7 Differences between `tabu`, `tabular`, `tabularx` and `longtable`

### 7.1 Paragraph indentation

See [Paragraph indentation](#)

### 7.2 Custom environments

Unlike `tabularx`, it is possible to define your own environment using `tabu`:

```
\newenvironment{foo}
  {\begin{tabu}{X[1.2]|[1pt gray]X}}
  {\end{tabu}}
```

`tabu` environment, even when `X` columns are used, may appear in the definition of your custom `tabular` environment.

You can also use the commands `\savetabu` `\preamble` (or `\usetabu`) for this purpose.

### 7.3 Inversion of tokens

When you typeset the following `tabular`:

```
\begin{tabular}{|>\bfseries>{ before }l<{ one }<{ two }|}
  cell content
\end{tabular}
```

You get the following result: before **cell content** **two** **one**

→ The word *before* is not bold, and **two** comes before **one**.

The reason is explained in the documentation of `array.sty`, and is related to the `array` environment in math mode when using `\newcolumntype`.

This rather strange inversion of tokens may be justified in math mode (otherwise, errors may occur) but not in text mode in our opinion. Inside a `tabu` environment, when not in math mode, the tokens are not reversed and you get the intuitively expected result:

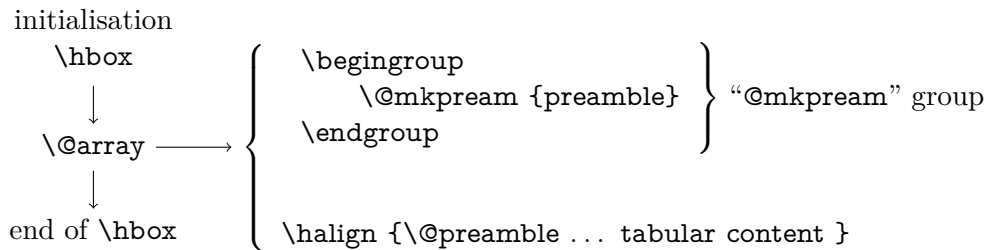
before cell content **one** **two**

In math mode however, tokens are in the reverse order in the `tabu` environment like they are in the `array` environment.



## 7.4 Improved process for rewriting columns (*for keen readers*)

Any tabular that does not split across pages is made with the following process:



For more details, see the [Flow chart of expansion](#).

`\@mkpream` works in two times inside a (semi-simple) group:

*First the rewriting process:*

Each special column in the tabular preamble is transformed into one the columns defined by `array.sty`.

*Second the building of the `\halign` preamble:*

The “rewritten preamble” is parsed and transformed in a preamble for the T<sub>E</sub>X primitive `\halign`. The result is stored into the `\@preamble` macro.

Any special columns of `tabu` are defined only inside the “`@mkpream`” group.

In the following example, you get an error with `tabular` and no error with `tabu`. With `tabular`, and `siunitx` `S` column, the *rewriting process* is as follow:

<pre> \documentclass {minimal} \usepackage {numprint,siunitx,xcolor} \usepackage {tabu} \begin {document}  \begin {tabular}{*2{S[color=green]}}   123,45 \end {tabular}  \begin {tabu}{*2{S[color=green]}}   123,45 \end {tabu}  \end {document} </pre>	<p>Inside <code>tabular</code>:</p> <ol style="list-style-type: none"> <li>1) Rewrite <code>S</code>: not found because inside <code>{...}</code></li> <li>2) Rewrite <code>*</code></li> <li>3) Rewrite <code>n</code> column defined by package <code>numprint</code> Then the ‘n’ in <code>green</code> is rewritten → problem</li> </ol> <p>Inside <code>tabu</code>:</p> <ol style="list-style-type: none"> <li>1) Rewrite <code>*</code></li> <li>2) Rewrite <code> </code> (there is none here)</li> </ol> <p>go back</p> <ol style="list-style-type: none"> <li>3) Rewrite <code>*</code></li> <li>4) Rewrite <code> </code></li> <li>5) Rewrite <code>S</code></li> <li>6) Rewrite <code>n</code> → not found because <code>S</code> was rewritten before, according to <code>siunitx</code> definition.</li> </ol>
---	--

The process of rewriting columns is usually longer inside `tabu` than inside `tabular`, but conversely `tabu` with `X` columns is optimised compared to `tabularx`, because the preamble is built only once, and not rebuilt before each trial as `tabularx` does. Thus `tabu` is much quicker than `tabularx`.

The process of rewriting is very sensitiv to the order in which columns are actually rewritten. This becomes critical when columns are defined with an optional argument like `tabu X` and `|` columns or `siunitx` `S` column.

## 8 The package options

### 8.1 The debugshow package option

 $\backslash\text{tracingtabu}$   
 $\backslash\text{tracingtabu} = 1, 2, 3 \text{ or } 4$ 

The control sequence  $\backslash\text{tracingtabu}$  has the same effect as the `debugshow` option:

- $\tau_{\text{N}}b\subset$  will report the widths it computes at each attempt to read the target, when `X` columns are used.
- Saved informations on the `tabu` are reported in the `.log` file when  $\backslash\text{savetabu}$  is used.

$\backslash\text{tracingtabu} = 2$  gives more information on the measures of the natural widths.

$\backslash\text{tracingtabu} = 3$  shows the struts inserted inside the `tabu` environment and gives more information about the measures of the height and depth of every row.

$\backslash\text{tracingtabu} = 4$  displays information on the insertions made by  $\backslash\text{tabucline}$ .

Typical information in the `.log` file:

(tabu) Try	tabu X	tabu Width	Target	Coefs	Update
(tabu) 1)	386.67296pt	797.34592pt	386.67296pt	2.0pt	-205.33649pt
(tabu) 2)	181.33647pt	386.67294pt	386.67296pt	2.0pt	0.00002pt
(tabu) 2)	Target reached (hfuzz=0.1pt) *****				

What does it mean?

- 1) The first attempt was performed with `X=386.67296pt`  
The `tabu` width (797.34592pt) exceeded the target by 410.67296pt.  
Thus `X` has been updated:  $410.67296\text{pt} / 2 = 205.33649\text{pt}$  and then:  
 $X = 386.67296\text{pt} - 205.33649\text{pt} = 181.33647\text{pt}$
- 2) The second attempt lead to a `tabu` width of 386.67294pt: the target is reached.  
The final width of each `X` column is the product of `tabu X` by its width coefficient.

### 8.2 The delarray package option

`delarray` option has the single effect to load `delarray.sty` for delimiters shortcuts around `tabu`. Delimiters shortcuts work both in math and text mode.

### 8.3 The linegoal package option

With the `linegoal` option, the default target for `tabu` with `X` columns is  $\backslash\text{linegoal}$  instead of  $\backslash\text{linewidth}$ . The `linegoal` package must be loaded and compilation must be done with pdf<sub>T</sub>E<sub>X</sub>, otherwise, a warning is displayed and the `linegoal` option has no effect: the default target remains  $\backslash\text{linewidth}$ .  $\backslash\text{linegoal}$  works with pdf<sub>T</sub>E<sub>X</sub> in pdf mode **and** in dvi mode.

If for some reason, you wish to turn down the `linegoal` option in your document, you can say (in a group for example):  $\backslash\text{let}\backslash\text{tabudefaultright}=\backslash\text{linewidth}$

In any case, specifying the target overwrites the default:  $\backslash\text{begin}\{\text{tabu}\}\text{ to}\backslash\text{linewidth}$

## 9 Corrections of some bugs (*available only inside tabu*)

### 9.1 Correction for colortbl and arydshln: compatibility with delarray

Both `colortbl` and `arydshln` forget the control sequence  $\backslash\text{@arrayright}$  in their implementation, quite strangely because both of them take care of  $\backslash\text{@arrayleft}$ . As a result, `delarray` shortcuts for delimiters around a tabular does not work if `colortbl` and/or `arydshln` are loaded.

Those control sequences are used by the `delarray` package to put variable size delimiters around the array:

<code>\begin {tabu} \{{X}.</code>		<code>\left \{\begin {tabu}{X}</code>
<code>...</code>	is like:	<code>...</code>
<code>\end {tabu}</code>		<code>\end {tabu} \right .</code>

## 9.2 Correction for `arydshln`: @ columns

A bug in `\adl@xarraydashrule`: `!-arg` columns (class 1) and `@-arg` columns (class 5) should be treated the same as far as rules are concerned.

With this correction, the “known problem number 1” in `arydshln` documentation is solved.

## 10 To do for even better `tabus`

In decreasing order of priority:

- Make double `\tabucline` compatible with `colortbl` `\doublerulesepcolor`
- Multiple `\tabucline` between different columns: extended specs:  
`\tabucline [line spec]{start-stop, start-stop}[line spec]{start-stop} ...`
- Reimplement `\multicolumn` in order to allow the `X` token in `\multicolumn` preamble.  
Provide `\multicell` to allow spanning columns and rows at the same time.
- Presently, `longtabu` with `X` columns works only if `\LTchunksize` is greater than the number of rows. I compiled a `longtabu` of 56 pages on my PC with `\LTchunksize = 2 000` without problem. Presently `\LTchunksize` is set to 10 000 during trials when `longtabu` contains `X` columns.
- Make `\tabucline` work with page breaks (one line on the top of the page, one line on the bottom of the previous).

# 11 Technical notice and Implementation

## 11.1 Drawing a tabular - The $\tau_{\text{nb}}\subset$ approach

$\tau_{\text{nb}}\subset$  has a different approach than almost any other package providing facilities for tabulars. `colortbl` and `arydshln` both put the cells contents into a box for measuring purpose, and then use the dimensions of each box to make their setups:

`colortbl` needs the dimensions of the box to put a rule in the background of the cell,

`arydshln` needs the dimensions to set the length of its leaders (dash lines).

This is achieved by modifying the macros defined in `array.sty` to insert columns inside the `\halign` preamble.

Instead,  $\tau_{\text{nb}}\subset$  proceeds as follow:

1. It first measures (if there are some negative width coefficients, or if `tabu spread` is used) the natural widths of the cells / the columns,
2. Then it always measures the height and depth of each cell / row,
3. Thereafter, the tabular is printed exactly as if `array.sty` was entitle to print it: no “extra” boxing of the cells material. The measurements have been stored and can be used to set the struts (only one per row) and the lengths of vertical leaders.
4. No macros of `array.sty` is modified at stage 3.

$\tau_{\text{nb}}\subset$  material inserted in the tabular for vertical leaders, `\rowfont` etc. is put inside the special “free” tokens provided by `array.sty`:

- A vertical leader is put inside a ! column: `!{vertical leader}`
- Changing font and alignment in one row requires some setup in > tokens: `>{rowfont material}`.

This way, the commands of `array.sty` that build each column definition (or preamble, in the sense of `\halign`) are never modified.

## 11.2 Algorithms

### `tabu to target`

The algorithm of `\tabu@arith` computes the desired widths to reach the target. In any case, only one measure of the tabular is required to get the widths for all columns. Here we describe the method with an example and some equations too to show that this handle all cases in generality.

**Notations and initialisation of X** In the case of `tabu to the target`  $T = 300$  is given : it is the target specified by the user or the default `tabu target` which is `\linewidth - \langle parindent correction \rangle` or `\linegoal`. Each X column has a width coefficient which is given too (or default to 1). The coefficients are:  $c_1, c_2, \dots, c_n$ .

X is the main dimension that drives the widths of all columns with a non negative coefficient, and limit the widths of columns with a negative coefficient.

Then we have first:

Coef $c_i$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$\sum$	$\Delta$
	-1	-2	-5	-2	2	3	15	
Target $T$	300							

Some coefficients are negative and we have to measure the natural widths of the corresponding columns, for columns always have a width:

$$\lambda_i = \begin{cases} c_i \cdot X & \text{if } c_i > 0 \\ \text{Min}(|c_i| \cdot X, \nu_i) & \text{if } c_i < 0 \end{cases} \quad \text{with} \quad \nu_i \leq T \quad \forall i$$

$\nu_i$  is the “natural width of the column” in the sense that it is the maximum of the natural widths of each cell in the  $i$ th X column, limited to the `tabu target`:  $\nu_i \leq T \quad \forall i$ .

$$\text{wd}(\text{table}) = \sum_i \lambda_i + \text{incompressible material} \left\{ \begin{array}{l} \bullet \text{ \code{\tabcolsep}} \\ \bullet \text{ vertical lines/leaders thickness} \\ \bullet \text{ non X columns} \end{array} \right.$$

So what is  $X$  at first ? Columns that have a non negative coefficients always have a width equal to  $\lambda_i = c_i \cdot X$  therefore, if we only have non negative coefficients, we can safely set:

$$X = \frac{T}{\sum_i c_i}$$

$$\begin{aligned} \forall i \quad c_i < 0 &\implies |c_i| \cdot X \geq T \\ \exists c_i > 0 &\implies \sum_{c_i > 0} c_i \cdot X \geq T \end{aligned}$$

And finally, for the measure:  $X = \text{Max} \left[ \text{Max}_{c_i < 0} \frac{T}{|c_i|}; \frac{T}{\sum_{c_i > 0} c_i} \right]$

Coef $c_i$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$\Sigma$	$\Delta$
	-1	-2	-5	-2	2	3	15	
Target $T$	300							
$X$	300							
$\nu_i$	10	300	80	80				
$\lambda_i$	10	300	80	80	600	900	1970	1800

We now choose a new value for  $X$ :

$$\sum_i \lambda_i = \sum_i \min_{c_i < 0} (\nu_i; c_i \cdot X) + \sum_{c_i > 0} c_i \cdot X \leq \sum_i |c_i| \cdot X$$

Let's try  $X' = \frac{\sum_i \lambda_i - \Delta}{\sum_i |c_i|}$  so that  $\sum_i \lambda'_i \leq \sum_i |c_i| \cdot X' \leq \sum_i \lambda_i - \Delta :$

Coef $c_i$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$\Sigma$	$\Delta$
	-1	-2	-5	-2	2	3	15	
Target $T$	300							
$X$	300							
$\nu_i$	10	300	80	80				
$\lambda_i$	10	300	80	80	600	900	1970	1800
$X'$	11.33							

$$X' = \frac{1970 - 1800}{15} = \frac{170}{15} = 11.33 \quad \text{Note that the computation of } X' \text{ does not involve any measurement.}$$

Coef $c_i$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$\sum$	$\Delta$
	-1	-2	-5	-2	2	3	15	
Target $T$	300							
$X$	300							
$\nu_i$	10	300	80	80				
$\lambda_i$	10	300	80	80	600	900	1970	1800
$X'$	11.33							
$\lambda'_i$	10,00	22,67	56,67	22,67	22,67	34,00	168,67	-1.33

Here we are in the case where the table width:

$$\begin{aligned} \text{wd}(\text{table}) &= \sum_i \lambda_i + \text{incompressible material} = T + \Delta \\ \Rightarrow \sum_i \lambda'_i + \text{incompressible material} &\leq \sum_i \lambda_i - \Delta + I = T \end{aligned}$$

Without any measure, we can say that the final table width will be less than the target, if we choose  $X'$ . The free space to share among the X columns (computed with  $X'$ ) is now  $\Delta' = T - (\sum_i \lambda'_i + I) = 300 - (168.67 + 130) = -1.33$ , where  $I$  is the incompressible material.

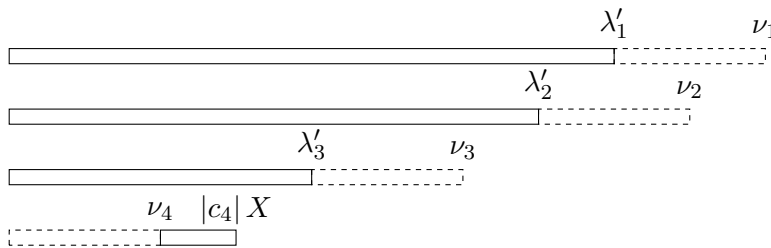
**Giving space** We say that a column is *saturated* (ie.full) if its natural width is greater than  $|c_i| \cdot X$ , or all the same that  $\lambda_i < \nu_i$ . We also will consider that the columns with  $c_i > 0$  have a “natural width” which is always equal to  $c_i \cdot X$ : in other words, a column with a non negative coefficient is always *saturated*.

Giving space (or “refunding” space) to the columns must be done in priority to the *saturated* columns. If all columns are finally underfull, then we will distribute the extra space to each, according a distribution rule. But this case can only occur if  $\forall i \quad c_i < 0$  because we first choosed  $X$  so that:

$$X \geq \frac{T}{\sum_{\substack{i \\ c_i > 0}} c_i}$$

and hence, the sum of the widths of the “non negative” columns exceeds the target.

Let’s rank the columns widths:



we first give space  
to the saturated columns  
1, 2 and 3

Because of the saturation, the total amount of space to give:  $|\Delta|$  shall be shared among the columns according to their widths coefficients. We shall not give too much space: the columns shall remain saturated. Let  $0 < \epsilon \leq |\Delta|'$  the amount of space to give, then after the operation:

$$\begin{aligned} \lambda_1'' + \lambda_2'' + \lambda_3'' &= \lambda'_1 + \lambda'_2 + \lambda'_3 + \epsilon \\ &= |c_1| X' + |c_2| X' + |c_3| X' + \epsilon \end{aligned}$$

Let’s say  $X'' = X' + \frac{|\Delta|'}{\sum_{\substack{i \\ c_i \text{ saturated}}} |c_i|}$  then it’s possible, without any measure, to compute:

$$\sum_{\substack{i \\ c_i \text{ saturated}}} \lambda''_i + \nu_4 \leq \sum_{\substack{i \\ c_i \text{ saturated}}} |c_i| \cdot X'' + \nu_4 \leq \sum_i |c_i| X' + \Delta' = \sum_i \lambda'_i + \Delta' \leq T - I$$

Or for clarity:  $\sum_i \lambda_i'' + I = \text{wd}(\text{table}) \leq T$  and the new free space to share is now :

$$\Delta'' = \left| T - \left( \sum_i \lambda_i'' + I \right) \right|$$

At each step of the computation, and without any measure but the first,  $X$  grows,  $\Delta$  decreases, and finally the target is reached for  $X$  such that  $\Delta \leq \text{hfuzz}$ .

Coef $c_i$	$c_1$	$c_2$	$c_3$	$c_4$	$c_5$	$c_6$	$\sum$	$\Delta$
	-1	-2	-5	-2	2	3	15	
Target $T$	300							
$X$	300							
$\nu_i$	10	300	80	80				
$\lambda_i$	10	300	80	80	600	900	1970	1800
$X'$	11.33							
$\lambda'_i$	10,00	22,67	56,67	22,67	22,67	34,00	168,67	-1.33
$X''$	11.43							
$\lambda''_i$	10,00	22,86	57,14	22,86	22,86	34,29	170,00	0

Now if the width of the table is less that the target, because 1) every column has a negative coefficient and 2) their natural widths are so small than the tabular don't fill the wanted horizontal space, the algorithm artificially raise the natural widths, according to a linear distribution:

$$\lambda'_i = \lambda_i + \Delta \cdot \frac{\lambda_i}{\sum_i \lambda_i} = \nu_i + \Delta \cdot \frac{\nu_i}{\sum_i \nu_i} = \nu_i \cdot \left( 1 + \frac{\Delta}{\sum_i \nu_i} \right)$$

## tabu spread dimen

The case of **tabu spread** is interesting and quite complex...

Here, the aim of the game is to give a target to the table, depending on its natural width. **tabu** has a default target (`\linewidth` in general, but it is possible to `\let \tabudefaulthtarget` to another value... for example `\linegoal`) which is a maximum for the final target of **tabu spread**. The case where the spread is *Opt* is not simpler nor more difficult.

If every column has a negative coefficient, it's rather easy because either the table exceeds the target, and then the new target will be the default target (the *maximum*), or the table width is less than the default target and we fix the new target to be that width + the spread, in the limit of the default target.

The condition that must hold on coefficient is not restritive if every column has a negative coefficient because if you say, for example:  $X = \text{Max}_i \frac{\nu_i}{|c_i|}$  then:

$$\sum_i \lambda_i = \sum_i \text{Min}(\nu_i; |c_i| \cdot X)$$

is true. It's always possible to find a  $X$  such that the behaviour anounced in the documentation is observed !

Then let's get some non negative coefficients. The natural widths of such columns must be measured, but the natural width of the tabular is not the same, for the proportions between column widths – expressed by their positive coefficient  $c_i$  – must be respected.

The real natural width of the tabular, which observe the proportions between columns with a non negative coefficient is:

$$\text{wd}(\text{table}) + \text{Max}_{\substack{i \\ c_i > 0}} \left( \frac{\nu_i}{c_i} \right) \times \sum_{\substack{i \\ c_i > 0}} c_i - \sum_{\substack{i \\ c_i > 0}} \nu_i > \text{wd}(\text{table})$$



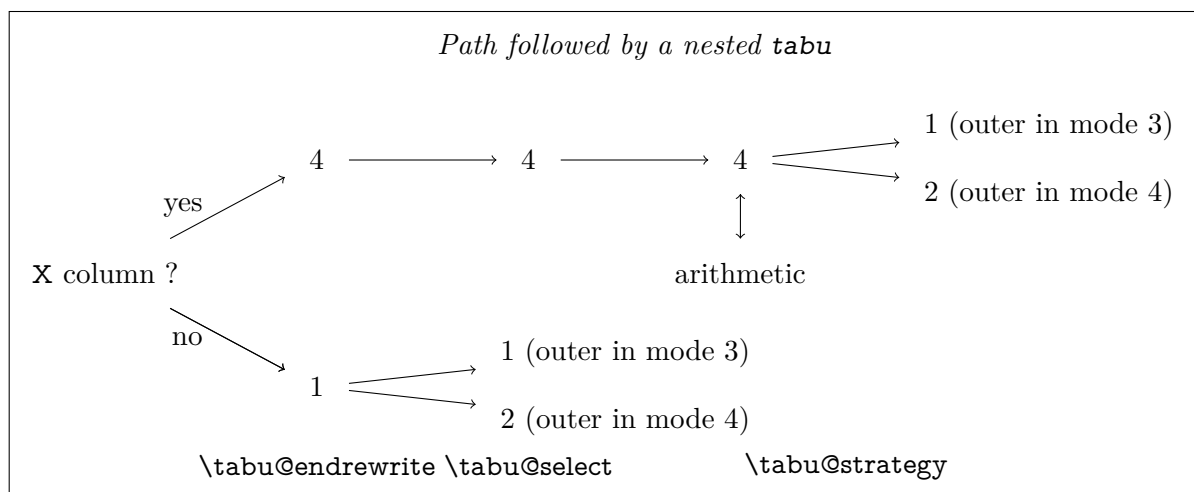
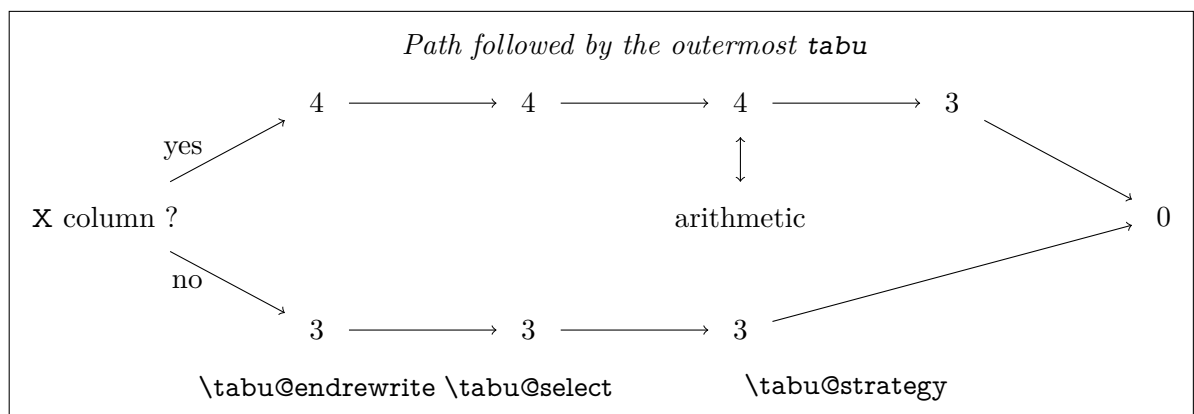
This quantity is computed,  $\tau_{nb}$  adds the **spread** and fix the new target to the sum, in the limit of the default target.

Then  $X$  is initialized such that:  $X = \text{Max} \left[ \text{Max}_{\substack{i \\ c_i < 0}} \frac{T}{|c_i|}; \frac{T}{\sum_{\substack{i \\ c_i > 0}} c_i} \right]$

and the algorithm described in the former section works, without any new measurement of the tabular. Unless this was not possible or deemed inconvenient for clarity, the code is presented in the same order it executes.

## 11.3 The tabu strategies

	Not nested (outer)	Nested
	$\backslash\text{count}@$ condition	$\backslash\text{count}@$ condition
$\backslash\text{tabu@endrewrite}$	3 no X column 4 X columns	0 outer is in mode 0 1 no X column 3 X column
$\backslash\text{tabu@select}$	3 or 4 needs trials 0 print out	0 outer in mode 0 $\Rightarrow$ print 1 outer in mode 3 2 from 1 in $\backslash\text{tabu@endrewrite}$ if outer in mode 4 3 or 4 needs trials
$\backslash\text{tabu@strategy}$	3 Vertical measure 4 Horizontal measure	1 Exit in vertical measure (outer in mode 3) 2 Exit with a rule (outer in mode 4) 4 Horizontal measure (nested in $\text{coef} < 0$ or spread)



## 11.4 Identification and Requirements

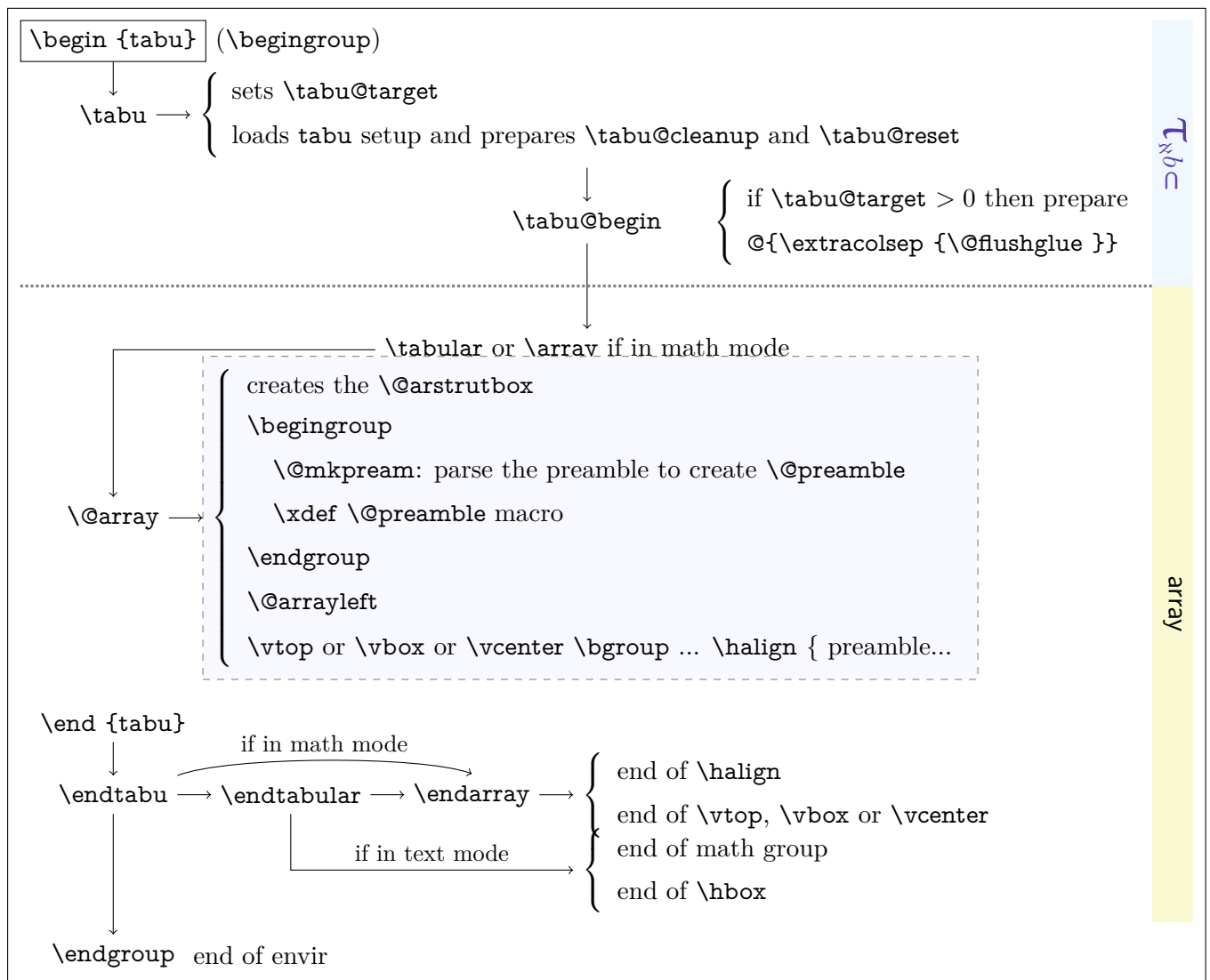
$\tau_{\text{N}b\text{C}}$  requires `array.sty` and `varwidth.sty`. The package namespace is `tabu@`.

```
1 \*package>
2 \NeedsTeXFormat{LaTeX2e}[2005/12/01]
3 \ProvidesPackage{tabu}[2011/02/19 v2.5 - flexible LaTeX tabulars (FC)]
4 \RequirePackage{array}[2008/09/09]
5 \RequirePackage{varwidth}[2009/03/30]
```

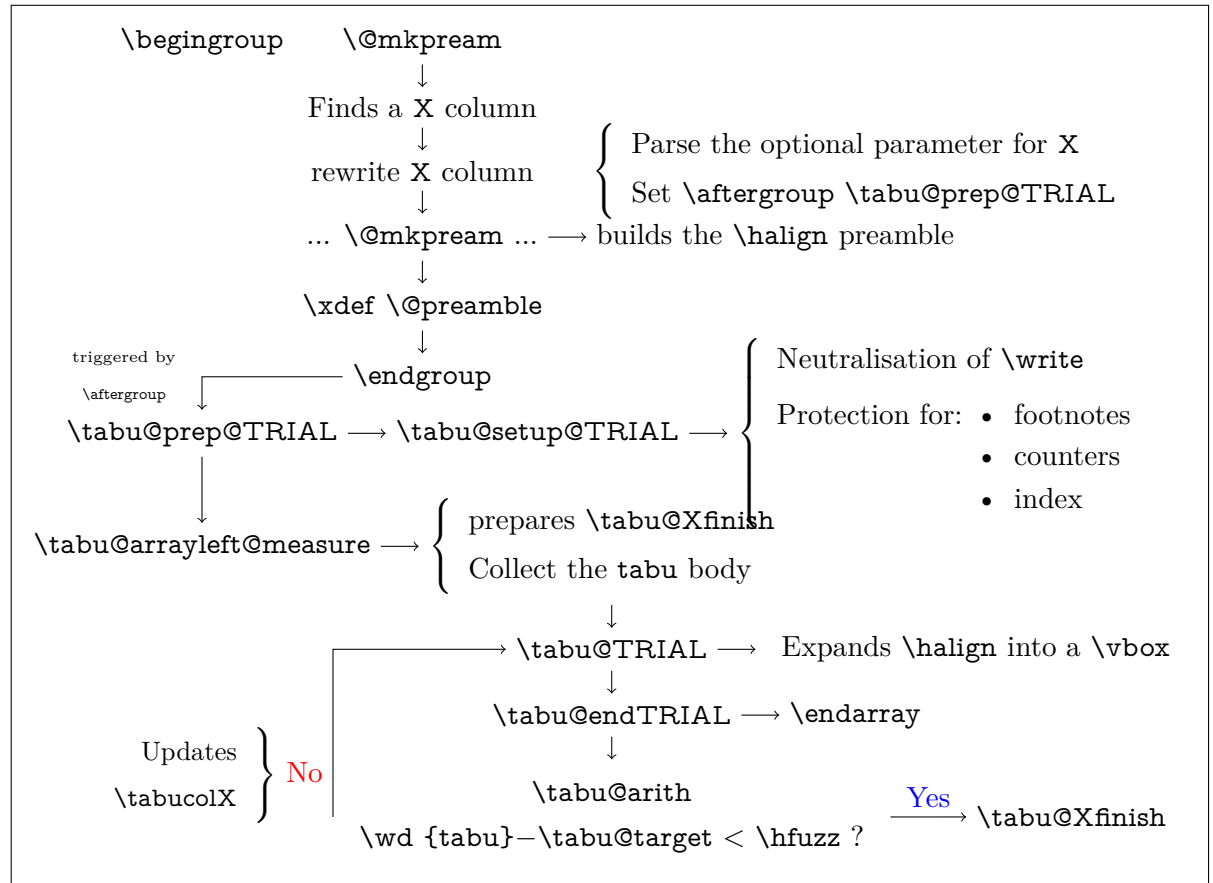
Minimal catcode accertaining for loading  $\tau_{\text{N}b\text{C}}$  in good conditions:

```
6 \AtEndOfPackage{\tabu@AtEnd \let\tabu@AtEnd \@undefined}
7 \let\tabu@AtEnd\@empty
8 \def\TMP@EnsureCode#1={%
9     \edef\tabu@AtEnd{\tabu@AtEnd
10         \catcode#1 \the\catcode#1}%
11     \catcode#1=%
12 }% \TMP@EnsureCode
13 \TMP@EnsureCode 33 = 12 % !
14 \TMP@EnsureCode 58 = 12 % : (for siunitx)
15 \TMP@EnsureCode 124 = 12 % |
16 \TMP@EnsureCode 36 = 3 % $ = math shift
17 \TMP@EnsureCode 38 = 4 % & = tab alignment character
18 \TMP@EnsureCode 32 = 10 % space
19 \TMP@EnsureCode 94 = 7 % ^
20 \TMP@EnsureCode 95 = 8 % _
```

## 11.5 Flow chart of expansion



**tabu to with X column** The important part of the job is made inside the dashed box above: `\@mkpream` expands the columns definitions, which can be user defined. Hopefully, it does its job inside a group, therefore a user-column can set a macro to be expanded `\aftergroup`. This implementation allows much modifications in the tabular preparation, without any change in the macros of `array.sty`.



## tabu spread with X column

In the case of “`tabu spread`” with `X` columns, the process is the same as the one described for “`tabu to`” with `X` columns. However, the first trial is different because we have first to measure the *natural width* of the tabular. The process is the following:

- `\tabu@target` is first set to `\linewidth` (or `\linegoal` with the `linegoal` package option).
- The `X` column corresponds to a `\vbox` with `\hsize` fixed to `\tabu@target`.
- Inside this `\vbox` the cell content is written into a `\hbox` whose width is limited to `\tabu@target`. This `\hbox` is captured into the box register `\tabu@box`.
- At the end of the cell, the `\badness` of the `\hbox` is checked:
  - if the `\badness` is  $> 1000$  then the text is too long and “`tabu spread`” is useless: `tabu to \tabu@target` give the same result.
  - Otherwise, we get the natural width of the cell content by: `\setbox \tabu@box \hbox {\unhbox \tabu @box}`
- At the end of the first trial, `\tabu@spreadarith` checks if:

$$\text{width}(\text{tabular}) + \text{spread} < \text{\linewidth (or \linegoal)}$$

- if not, then `tabu to \tabu@target` give the same result
- Otherwise, the target for `tabu to` will be:

$$\text{width}(\text{tabular}) + \text{spread} - \underbrace{\sum_i \text{natural widths } X_i + \text{Max}_i \left( \frac{\text{natural width } X_i}{\text{coef}_i} \right) \times \sum_i \text{coef}_i}_{\text{minimal natural width that can be obtained with the given coefs}}$$

And the next trial will be done as if the user called “`tabu to`” with this target.

## 11.6 Some constants

Here we define the constants used by  $\mathcal{T}_{\text{NB}}$ :  $\text{\TeX}$  registers and a few *helper* macros.

When working inside a tabular (*ie.*  $\text{\halign}$ ) each cell is a  $\text{\TeX}$  group. Probably the most important property of each register defined here is whether it is global or not. A *local* register does not suffer, never, any global assignment.

### $\text{\TeX}$ registers

**taburow**  $\text{\LaTeX}$  counter that globally stores the value of the current row. It is updated at  $\text{\everyrow}$ , rather than at  $\text{\everycr}$ <sup>8</sup>.  $\text{\thetaburow}$  expands to the (arabic) number.

This counter can be read by the user, but she **must not change its value** because it is used internally to store the height/depth of every row, for vertical spacing adjustment (and vertical leaders).

**\tabu@nbcols**  $\text{\TeX}$  counter that – locally – saves the total number of columns of the **tabu**. Special **@** and **!** columns are not counted (they are not *real* columns for  $\text{\halign}$ , but only insertions into the preamble).

The value is used by  $\text{\tabucline}$  to ensure that the leader does not jut out over the last column...

**\tabu@cnt**  $\text{\TeX}$  counter that – locally – stores the number of trials. Incidentally, it is also temporarily used to parse the width coefficient for **X** columns, during the rewriting process.

**\tabu@Xcol**  $\text{\TeX}$  counter that – locally – stores the number of **tabu** **X** columns. Defined while rewriting the **X** token, it is used in the specification of the width of the column ( $\text{\tabu@hsize}$  {Rank of the **X** column}{coef}).

It is also used to store the natural width of **X** columns (in the cases of a negativ coefficient or if **tabu spread** is used).

**\tabu@alloc** A global counter whose initial value ( $-1$ ) is incremented for each nested tabular. The end of the outermost tabular globally resets the value to  $-1$ .  $\text{\tabu@nested}$  stores locally the value of  $\text{\tabu@alloc}$  and is therefore the “index” of the current tabular (the one that is actually in construction).

This influences the initialisation process (cf.  $\text{\tabu@setup}$  and  $\text{\tabu@init}$ ).

**\tabu@start** **\tabu@stop** They are used locally by  $\text{\tabucline}$  and  $\text{\everyrow}$  while parsing the parameters: this is, for clarity, the local name for  $\text{\@tempcnta}$  and  $\text{\@tempcntb}$ .

```
21 \newcount \c@taburow      \def\thetaburow {\number\c@taburow}
22 \newcount \tabu@nbcols
23 \newcount \tabu@cnt
24 \newcount \tabu@Xcol
25 \let\tabu@start \@tempcnta
26 \let\tabu@stop  \@tempcntb
27 \newcount \tabu@alloc    \tabu@alloc=\m@ne
28 \newcount \tabu@nested
29 \def\tabu@alloc@{\global\advance\tabu@alloc \@ne \tabu@nested\tabu@alloc}
```

**\tabu@target**  $\text{\TeX}$  dimen that – locally – stores the **tabu** target (either “to” or “spread”).

**\tabu@spreadtarget**  $\text{\TeX}$  dimen that – locally – stores the **tabu** spread given by the user.

**\tabu@naturalX**  $\text{\TeX}$  dimen that – globally – stores the total natural widths of the **X** columns, in the cases of negativ coefficients and/or **tabu spread**. The value is reset to *Opt* at  $\text{\everyrow}$ , and *maxima/minima* are stored into the macro  $\text{\tabu@naturalXmin}$  and  $\text{\tabu@naturalXmax}$ : those are required for the algorithm of **tabu spread** ( $\text{\tabu@spreadarith}$ ).

**\tabucolX**  $\text{\TeX}$  dimen that – locally – stores the width corresponding to the preamble token **X**[1]: the standard width of **X** columns.

**\tabu@DELTA** This is for clarity, the local name of  $\text{\@tempdimc}$  in  $\text{\tabu@arith}$ .

**\tabu@thick** **\tabu@on** **\tabu@off** They are used locally by  $\text{\tabu@getline}$ , while parsing the parameters for a line specification. This is for clarity, the local name for  $\text{\@tempdima}$ ,  $\text{\@tempdimb}$  and  $\text{\@tempdimc}$ .

8. Package **xcolor** defines the **\rownum**  $\text{\TeX}$  counter, which is globally updated at  $\text{\everycr}$ . Hence this **\rownum** counter is not reliable in case the user invokes  $\text{\cline}$  or  $\text{\cmidrule}$  for example...

```
30 \newdimen \tabu@target
31 \newdimen \tabu@spreadtarget
32 \newdimen \tabu@naturalX
33 \newdimen \tabu@colX
34 \let\tabu@DELTA \@tempdimc
35 \let\tabu@thick \@tempdima
36 \let\tabu@on \@tempdimb
37 \let\tabu@off \@tempdimc
```

**\tabu@Xsum** T<sub>E</sub>X dimen that – locally – stores the sum of all width coefficients for X columns. This is required to fix the initial value for **\tabu@colX** and then in the algorithms (**\tabu@arith** and **\tabu@arithnegcoef**).

```
38 \newdimen \tabu@Xsum
```

**\extrarowdepth** **array.sty** defines **\extrarowheight** as a T<sub>E</sub>X dimen register: the extra height to be finally added to each row of a table.  $\tau_{\text{NB}} \subset$  defines **\extrarowdepth** in addition: the *extra depth*. Though **\extrarowheight** and **\extrarowdepth** can be set by the user, the official interface is **\extrarowsep**.

**\abovetabulinesep** T<sub>E</sub>X dimensions **\abovetabulinesep** and **\belowtabulinesep** store the minimum allowed vertical space between the contents of the cells and their borders. Their values are ignored if non positive. Though they can be set by the user, the official interface is **\tabulinesep**.

The philosophy and the technics are similar to the one provided by the **cellspace** package. However, limitations of **cellspace** are lifted (nested **tabu** environments, use of colors... see the **cellspace limitations** in the revision history).  $\tau_{\text{NB}} \subset$  inserts only one strut per line, whose name is **\@arstrut**.

**\tabustrutrule** The T<sub>E</sub>X dimen **\tabustrutrule** is here only for debugging purpose: its value must be *0pt*. It behaves mostly like T<sub>E</sub>X primitive **\overfullrule**, and allow to see the struts introduced in the tabular, and to control vertical spacing. Setting **\tabustrutrule** to a positive value has no effect unless **\tracingtabu** is  $\geq 3$ . The official interface is **\tracingtabu = 3**.

```
39 \newdimen \extrarowdepth
40 \newdimen \abovetabulinesep
41 \newdimen \belowtabulinesep
42 \newdimen \tabustrutrule \tabustrutrule \z@
```

**\tabu@thebody** This token stores – locally – the collected content of the **tabu** environment during the measuring process.

**\tabu@footnotes** Token that globally stores the footnotes inside the **tabu** environment, for **\insert** does not work inside such a level of groupings...

```
43 \newtoks \tabu@thebody
44 \newtoks \tabu@footnotes
```

**\tabu@box** Stores – locally – the whole **tabu** when an attempt to adjust X columns is performed.

**\tabu@arstrutbox** While the **\@arstrutbox** may be redefined globally at the end of each line (for vertical spacing adjustment), we define a new box and **\let \@arstrutbox** to be that box inside the **tabu** environment.

Hence, the **\@arstrutbox** used by other tabular environment does not suffer any modification.

**\tabu@hleads** Those boxes are used to build horizontal and vertical leaders. In order not to rebuild the boxes every time a leader is inserted, the box is globally defined if a line style is specified (*via* **|line style** or **\tabucline** **[line style]{...}** or **\tabulinestyle** **{line style}**).

```
45 \newsavebox \tabu@box
46 \newsavebox \tabu@arstrutbox
47 \newsavebox \tabu@hleads
48 \newsavebox \tabu@vleads
```

## Switches

<code>\iftabu@colortbl</code>	The global switch <code>\iftabu@colortbl</code> is used by <code>\rowfont</code> when modifying the alignments, because <code>colortbl</code> changes the glues put inside the <code>\halign</code> preamble to make standard alignments. This switch is set <code>AtBegin Document</code> .
<code>\iftabu@siunitx</code>	Global switch set <code>\AtBeginDocument</code> . <code>true</code> if <code>siunitx</code> package is detected.
<code>\iftabu@measuring</code>	This switch is somewhat <i>magic</i> in the sense that it has several meanings... It is temporarily set to <code>true</code> by <code>\tabu@arith</code> in the trial group, to say that the <code>tabu</code> did not reach its target yet. It is also set to <code>true</code> in the <code>\@mkpream</code> group when the first <code>X</code> column is encountered in the preamble. Finally, it is true in the <code>trialS</code> group when the outermost tabular is in strategy number 2 or number 3.
<code>\iftabu@spread</code>	A switch whether “ <code>tabu spread</code> ” is used or not. A nested <code>tabu</code> inside a <code>X</code> column whose coefficient is negative has a default target set to <code>spread 0pt</code> .
<code>\iftabu@negcoef</code>	A switch set to true in case of negativ coef (natural width if less than <code>X[coef]</code> ).
<code>\iftabu@everyrow</code>	A very important global switch: <code>true</code> when outside any <code>tabu</code> environment, <code>true</code> as well when inside a cell of a <code>tabu</code> , but globally set to <code>false</code> at <code>\everycr</code> and therefore inside any <code>\noalign</code> command. This allows to insert leaders (by <code>\omit \span \omit \cr \noalign {...}</code> ) or first/last line corrections only once, even if <code>\everycr</code> is executed more than once.
<code>\iftabu@long</code>	Finally the swith <code>\iftabu@long</code> is set to <code>true</code> inside <code>longtabu</code> and to <code>false</code> inside <code>tabu</code> . This is convenient because some setup are slightly different between <code>tabu</code> and <code>longtabu</code> .

```

49 \newif \iftabu@colortbl
50 \newif \iftabu@siunitx
51 \newif \iftabu@measuring
52 \newif \iftabu@spread
53 \newif \iftabu@negcoef
54 \newif \iftabu@everyrow
55 \def\tabu@everyrowtrue {\global\let\iftabu@everyrow \iftrue}
56 \def\tabu@everyrowfalse{\global\let\iftabu@everyrow \iffalse}
57 \newif \iftabu@long

```

<code>\iftabuscantokens</code>	<code>\iftabuscantokens</code> is the switch for whether or not <code>tabu</code> will use <code>\scantokens</code> . Though the user can set <code>\iftabuscantokens</code> to <code>\iftrue</code> or <code>\iffalse</code> , the official interface is <code>tabu*</code> .
--------------------------------	--

`\tabu@rescan`

It does not make sense to use `\scantokens` in a nested `tabu`: only the outermost `tabu` can use `\scantokens`, for the environment body must be collected with care !

`\tabu@rescan` is the helper macro for scanning tokens.

```

58 \newif \iftabuscantokens
59 \def\tabu@rescan {\tabu@verbatim \scantokens }

```

## Some helper macros

<code>\tabu@gobblespace</code>	Two macros which are needed when scanning tokens with <code>\futurelet</code> .
<code>\tabu@gobbletoken</code> <code>\tabu@gobblex</code>	This gobbles the character number 10 in ASCII ( <code>^^J</code> in T <sub>E</sub> X).
<code>\tabu@ifenvir</code>	Checks if the current environment is <code>tabu</code> or <code>longtabu</code> (for <code>\multicolumn</code> inside <code>tabu</code> ).
<code>\tabu@modulo</code>	Computes the modulo (for <code>\taburowcolors</code> ). The method is taken from H.O. <code>intcalc</code> package.

```

60 \def\tabu@gobblespace #1 {#1}
61 \def\tabu@gobbletoken #1#2{#1}
62 \def\tabu@gobblex{\futurelet\@let@token \tabu@gobblex}
63 \def\tabu@gobblex{\if ^^J\noexpand\@let@token \expandafter\@gobble
64 \else\ifx \@sptoken\@let@token
65 \expandafter\tabu@gobblespace\expandafter\tabu@gobblex
66 \fi\fi
67 }% \tabu@gobblex
68 \def\tabu@X{^^J}

```

```

69 {\obeyspaces
70 \global\let\tabu@spxiii= % saves an active space (for \ifx)
71 \gdef\tabu@@spxiii{ }}
72 \def\tabu@ifenvir {% only for \multicolumn
73     \expandafter\tabu@if@nvir\csname \@currentenv\endcsname
74 }% \tabu@ifenvir
75 \def\tabu@if@nvir #1{\csname @\ifx\tabu#1first\else
76                     \ifx\longtabu#1first\else
77                     second\fi\fi oftwo\endcsname
78 }% \tabu@ifenvir
79 \def\tabu@modulo #1#2{\numexpr\ifnum\numexpr#1=\z@ 0\else #1-(#1-(#2-1)/2)/(#2)*(#2)\fi}

```

**\tabu@strtrim** Trimming spaces at low cost...

```

80 {\catcode`\&=3
81 \gdef\tabu@strtrim #1{% #1 = control sequence to trim
82     \ifodd 1\ifx #1\@empty \else \ifx #1\space \else 0\fi \fi
83     \let\tabu@c@l@r \@empty \let#1\@empty
84     \else \expandafter \tabu@trimspaces #1&#1\@nnil
85     \fi
86 }% \tabu@strtrim
87 \gdef\tabu@trimspaces #1&#2\@nnil{\let\tabu@c@l@r=#2\tabu@firstspace .#1& &#2}%
88 \gdef\tabu@firstspace #1#2#3 &{\tabu@lastspace #2#3&}
89 \gdef\tabu@lastspace #1&#2&#3{\def #3{#1}%
90     \ifx #3\tabu@c@l@r \def\tabu@c@l@r{\protect\color{#1}}\expandafter\remove@to@nnil \fi
91     \tabu@trimspaces #1&#3\@nnil}
92 }% \catcode

```

**\tabu@sanitizearg** Sanitize an argument (babel compliant).

```

93 \def\tabu@sanitizearg #1#2{%
94     \csname \ifcsname if@safe@actives\endcsname % <babel>
95         @safe@activetrue\else
96         relax\fi \endcsname
97     \edef#2{#1}\tabu@strtrim#2\@onelevel@sanitize#2%
98     \expandafter\expandafter\def\expandafter#2\expandafter{#2}%
99 }% \tabu@sanitizearg

```

**\tabu@textbar** The character | may have a special category code inside the document, depending on the language setting or for example, | can be the delimiter shortcut for verbatim. We use \scantokens to allow an \ifx test even if the category code of | changes along the compilation.

```

100 \def\tabu@textbar #1{\begingroup \endlinechar\m@ne \scantokens{\def\:{|}}%
101     \expandafter\endgroup \expandafter#1\:% !!! semi simple group !!!
102 }% \tabu@textbar

```

**\tabu@everyrow@bgroup** Commands like \everyrow, \taburulecolor, \tabulinestyle, \taburowcolors can be expanded either in a cell or outside a tabu environment or at the end of a row, inside a \noalign group.

To avoid the insertion of an empty math atom (equivalent to \hbox to0pt{ }) we open a semi-simple group rather than a math group if not in \noalign. \toks@ is used to define the local-to-the- $\TeX$ -group setting (post-fixed by @L).

```

103 \def\tabu@everyrow@bgroup{\iftabu@everyrow \begingroup \else \noalign{\ifnum0='}\fi \fi}
104 \def\tabu@everyrow@egroup{%
105     \iftabu@everyrow \expandafter \endgroup \the\toks@
106     \else \ifnum0='{}\fi}%
107     \fi
108 }% \tabu@everyrow@egroup

```



## Rebuild the \@arstrutbox

**\tabu@arstrut** The macros rebuilds the \@arstrutbox (a \hbox). With the *debug* variants when \tracingtabu = 3 and \tabustrutrul> 0.

```
109 \def\tabu@arstrut {\global\setbox\@arstrutbox \hbox{\vrule
110   height \arraystretch \dimexpr\ht\strutbox+\extrarowheight
111   depth \arraystretch \dimexpr\dp\strutbox+\extrarowdepth
112   width \z@}%
113 }% \tabu@arstrut
114 \def\tabu@rearstrut {%
115   \@tempdima \arraystretch\dimexpr\ht\strutbox+\extrarowheight \relax
116   \@tempdimb \arraystretch\dimexpr\dp\strutbox+\extrarowdepth \relax
117   \ifodd 1\ifdim \ht\@arstrutbox=\@tempdima
118     \ifdim \dp\@arstrutbox=\@tempdimb 0 \fi\fi
119   \tabu@mkarstrut
120   \fi
121 }% \tabu@rearstrut
```

**\tabu@DBG@arstrut** This is the “debug” version of \tabu@arstrut: used when \tracingtabu = 3 or more to show the struts inserted in the tabular.

```
122 \def\tabu@DBG #1{\ifdim\tabustrutrul>\z@ \color{#1}\fi}
123 \def\tabu@DBG@arstrut {\global\setbox\@arstrutbox
124   \hbox to\z@{\hbox to\z@{\hss
125     {\tabu@DBG{cyan}\vrule
126     height \arraystretch \dimexpr\ht\strutbox+\extrarowheight
127     depth \z@
128     width \tabustrutrul}\kern-\tabustrutrul
129     {\tabu@DBG{pink}\vrule
130     height \z@
131     depth \arraystretch \dimexpr\dp\strutbox+\extrarowdepth
132     width \tabustrutrul}}}%
133 }% \tabu@DBG@arstrut
```

**\tabu@save@decl** No inversion on tokens in the tabu preamble, when not in math mode.

```
134 \def\tabu@save@decl{\toks\count@ \expandafter{\the\toks\expandafter\count@
135   \nextchar}}%
136 \def\tabu@savdecl{\ifcat$\d\llarend\else
137   \let\save@decl \tabu@save@decl \fi % no inversion of tokens in text mode
138 }% \tabu@savdecl
```

**\tabu@finalstrut**

```
139 \def\tabu@finalstrut #1{\unskip\ifhmode\nobreak\fi\vrule height\z@ depth\z@ width\z@}
```

## Disable some commands during trials

**\tabuDisableCommands** Following the model of hyperref \pdfstringdefDisableCommands, \tabuDisableCommands allow the user to change the definition of some commands during the trial loops, by the mean of a hook to be expanded by \tabu@setstrategy.

```
140 \newcommand*\tabuDisableCommands {\g@addto@macro\tabu@trialh@@k }
141 \let\tabu@trialh@@k \empty
```

**\tabu@nowrite** A trick (from the T<sub>E</sub>X-book) to forbidd \write when a trial is done on the \halign.

**\tabu@nofootnotes** Disable footnotes during trials.

```
142 \def\tabu@nowrite #1#{{\afterassignment}\toks@}
143 \let\tabu@write\write
144 \let\tabu@immediate\immediate
145 \def\tabu@WRITE{\begingroup
146   \def\immediate\write{\aftergroup\endgroup
```

```

147          \tabu@immediate\tabu@write}%
148 }% \tabu@WRITE
149 \expandafter\def\expandafter\tabu@GenericError\expandafter{%
150          \expandafter\tabu@WRITE\GenericError}
151 \def\tabu@warn{\tabu@WRITE\PackageWarning\tabu}
152 \def\tabu@noxfootnote [#1]{\@gobble}

```

**\tabu@nocolor** For optimisation purpose, color changes are deactivated during trials, for they do not affect the measures.

```

\tabu@norowcolor 153 \def\tabu@nocolor #1#{\@gobble}
154 \newcommand*\tabu@norowcolor[2][{}

```

## siunitx S and s columns management

**\tabu@maybesiunitx** A macro that encloses the definition of `\tabu@celllalign`, in order to check if the column is a siunitx S (or s) column, and neutralise the setup of `\rowfont` in this case, for siunitx provides its own `key=value` options to set fonts inside S (or s) columns.

```

155 \def\tabu@maybesiunitx #1{\def\tabu@temp{#1}%
156          \futurelet\@let@token \tabu@m@ybesiunitx}
157 \def\tabu@m@ybesiunitx #1{\def\tabu@m@ybesiunitx {%
158     \ifx #1\@let@token \let\tabu@cellleft \@empty \let\tabu@cellright \@empty \fi
159     \tabu@temp}% \tabu@m@ybesiunitx
160 }\expandafter\tabu@m@ybesiunitx \csname siunitx_table_collect_begin:Nn\endcsname
161 \def\tabu@celllalign\def #1{\def\tabu@celllalign{\tabu@maybesiunitx{#1}}}%

```

## 11.7 Rules, colors and vertical adjustment

### `\extrarowsep` and `\tabulinesep`

**\extrarowsep** `\extrarowsep` makes the assignment for both `\extrarowheight` and `\extrarowdepth`.

The macro may be prefixed by `\global`.

```

162 \newcommand*\extrarowsep{\edef\tabu@C@extra{\the\numexpr\tabu@C@extra+1}%
163     \iftabu@everyrow \aftergroup\tabu@Gextra
164     \else \aftergroup\tabu@n@Gextra
165     \fi
166     \@ifnextchar={\tabu@gobbletoken\tabu@extra} \tabu@extra
167 }% \extrarowsep
168 \def\tabu@extra {\@ifnextchar_%
169     {\tabu@gobbletoken{\tabu@setextra\extrarowheight \extrarowdepth}}
170     {\ifx ^\@let@token \def\tabu@temp{%
171         \tabu@gobbletoken{\tabu@setextra\extrarowdepth \extrarowheight}}%
172     \else \let\tabu@temp \@empty
173         \afterassignment \tabu@setextrasep \extrarowdepth
174     \fi \tabu@temp}%
175 }% \tabu@extra
176 \def\tabu@setextra #1#2{\def\tabu@temp{\tabu@extr@#1#2}\afterassignment\tabu@temp#2}
177 \def\tabu@extr@ #1#2{\@ifnextchar^%
178     {\tabu@gobbletoken{\tabu@setextra\extrarowdepth \extrarowheight}}
179     {\ifx _\@let@token \def\tabu@temp{%
180         \tabu@gobbletoken{\tabu@setextra\extrarowheight \extrarowdepth}}%
181     \else \let\tabu@temp \@empty
182         \tabu@Gsave \tabu@G@extra \tabu@C@extra \extrarowheight \extrarowdepth
183     \fi \tabu@temp}%
184 }% \tabu@extr@
185 \def\tabu@setextrasep {\extrarowheight=\extrarowdepth
186     \tabu@Gsave \tabu@G@extra \tabu@C@extra \extrarowheight \extrarowdepth
187 }% \tabu@setextrasep
188 \def\tabu@Gextra{\ifx \tabu@G@extra\@empty \else {\tabu@Rextra}\fi}
189 \def\tabu@n@Gextra{\ifx \tabu@G@extra\@empty \else \noalign{\tabu@Rextra}\fi}

```

```
190 \def\tabu@Rextra{\tabu@Grestore \tabu@G@extra \tabu@C@extra}
191 \let\tabu@C@extra \z@
192 \let\tabu@G@extra \@empty
```

**\tabulinesep** **\tabulinesep** makes the assignment for both **\abovetabulinesep** and **\belowtabulinesep**.

The macro may be prefixed by **\global**.

```
193 \newcommand*\tabulinesep{\edef\tabu@C@linesep{\the\numexpr\tabu@C@linesep+1}%
194   \iftabu@everyrow   \aftergroup\tabu@Glinesep
195   \else               \aftergroup\tabu@n@Glinesep
196   \fi
197   \@ifnextchar={\tabu@gobbletoken\tabu@linesep} \tabu@linesep
198 }% \tabulinesep
199 \def\tabu@linesep {\@ifnextchar_%
200   {\tabu@gobbletoken{\tabu@setsep\abovetabulinesep \belowtabulinesep}}
201   {\ifx ^\@let@token \def\tabu@temp{%
202     \tabu@gobbletoken{\tabu@setsep\belowtabulinesep \abovetabulinesep}}}%
203   \else \let\tabu@temp \@empty
204     \afterassignment \tabu@setlinesep \abovetabulinesep
205   \fi \tabu@temp}%
206 }% \tabu@linesep
207 \def\tabu@setsep #1#2{\def\tabu@temp{\tabu@sets@p#1#2}\afterassignment\tabu@temp#2}
208 \def\tabu@sets@p #1#2{\@ifnextchar^%
209   {\tabu@gobbletoken{\tabu@setsep\belowtabulinesep \abovetabulinesep}}
210   {\ifx _\@let@token \def\tabu@temp{%
211     \tabu@gobbletoken{\tabu@setsep\abovetabulinesep \belowtabulinesep}}}%
212   \else \let\tabu@temp \@empty
213     \tabu@Gsave \tabu@G@linesep \tabu@C@linesep \abovetabulinesep \belowtabulinesep
214   \fi \tabu@temp}%
215 }% \tabu@sets@p
216 \def\tabu@setlinesep {\belowtabulinesep=\abovetabulinesep
217   \tabu@Gsave \tabu@G@linesep \tabu@C@linesep \abovetabulinesep \belowtabulinesep
218 }% \tabu@setlinesep
219 \def\tabu@Glinesep{\ifx \tabu@G@linesep\@empty \else {\tabu@Rlinesep}\fi}
220 \def\tabu@n@Glinesep{\ifx \tabu@G@linesep\@empty \else \noalign{\tabu@Rlinesep}\fi}
221 \def\tabu@Rlinesep{\tabu@Grestore \tabu@G@linesep \tabu@C@linesep}
222 \let\tabu@C@linesep \z@
223 \let\tabu@G@linesep \@empty
```

**\tabu@Gsave** Utility macros to implement the possibility to prefix a macro by **\global**.

**\tabu@Grestore**

```
224 \def\tabu@Gsave #1#2#3#4{\xdef#1{#1%
225   \toks#2{\toks\the\currentgrouplevel{\global#3\the#3\global#4\the#4}}}%
226 }% \tabu@Gsave
227 \def\tabu@Grestore#1#2{%\ifdim \baselineskip=\z@\noalign\fi
228   \toks#2{#1\toks\currentgrouplevel\expandafter{\expandafter}\the\toks#2\relax
229   \ifcat$\the\toks\currentgrouplevel$\else
230     \global\let#1\@empty \global\let#2\z@
231     \the\toks\currentgrouplevel
232   \fi
233 }% \tabu@Grestore
```

## Setting code for every row

**\everyrow** As long as `tabu` needs to execute some code at `\everycr`, it's not difficult to provide a command to give the user the opportunity to execute its own arbitrary code. However, `\everyrow` will be used almost only with `\hline` (or `\tabucline` or `\midrule`).

`\everyrow` can be changed anywhere inside the `tabu`: at the end of a row, or even inside a cell.

The rows L<sup>A</sup>T<sub>E</sub>X counter `taburow` must not be changed by the user!.

The settings are saved in a “locally-global” way...

```

234 \newcommand*\everyrow{\tabu@everyrow@bgroup
235           \tabu@start \z@ \tabu@stop \z@ \tabu@evrstartstop
236 }% \everyrow
237 \def\tabu@evrstartstop {\@ifnextchar^%
238   {\afterassignment \tabu@evrstartstop \tabu@stop=}%
239   {\ifx ^\@let@token
240     \afterassignment\tabu@evrstartstop \tabu@start=%
241     \else \afterassignment\tabu@everyr@w \toks@
242     \fi}%
243 }% \tabu@evrstartstop
244 \def\tabu@everyr@w {%
245   \xdef\tabu@everyrow{%
246     \noexpand\tabu@everyrowfalse
247     \let\noalign \relax
248     \noexpand\tabu@rowfontreset
249     \iftabu@colortbl \noexpand\tabu@rc@ \fi % \taburowcolors
250     \let\noexpand\tabu@docline \noexpand\tabu@docline@evr
251     \the\toks@
252     \noexpand\tabu@evrh@@k
253     \noexpand\tabu@rearstrut
254     \global\advance\c@taburow \@ne}%
255   \iftabu@everyrow \toks@\expandafter
256     {\expandafter\def\expandafter\tabu@evr@L\expandafter{\the\toks@}}%
257   \else \xdef\tabu@evr@G{\the\toks@}%
258   \fi
259   \tabu@everyrow@egroup
260 }% \tabu@everyr@w
261 \def\tabu@evr {\def\tabu@evrh@@k} % for internal use only
262 \tabu@evr{}

```

## Setting line styles and colors

**\newtabulinestyle** `\newtabulinestyle {style=spec.,style=spec,style=spec}`

All the job is done by `\tabu@getline`. New line style specification are always defined globally, and can be overwritten without warning...

```

263 \newcommand*\newtabulinestyle [1]{%
264   {\@for \@tempa :=#1\do{\expandafter\tabu@newlinestyle \@tempa==\@nil}}%
265 }% \newtabulinestyle
266 \def\tabu@newlinestyle #1=#2=#3\@nil{\tabu@getline {#2}%
267   \tabu@sanitizarg {#1}\@tempa
268   \ifodd 1\ifx \@tempa\@empty \ifdefined\tabu@linestyle@ 0 \fi\fi
269   \global\expandafter\let
270     \csname tabu@linestyle@\@tempa \endcsname =\tabu@thestyle \fi
271 }% \tabu@newlinestyle

```

**\tabulinestyle** `\tabulinestyle {style name}` or `\tabulinestyle line specs / leader`

The job is done by `\tabu@getline`. The settings as usual, are stored in a “locally-global” way...

```

272 \newcommand*\tabulinestyle [1]{\tabu@everyrow@bgroup \tabu@getline{#1}%

```

```

273 \iftabu@everyrow
274 \toks@{\expandafter{\expandafter \def \expandafter
275 \tabu@ls@L\expandafter{\tabu@thestyle}}}%
276 \gdef\tabu@ls@{\tabu@ls@L}%
277 \else
278 \global\let\tabu@ls@G \tabu@thestyle
279 \gdef\tabu@ls@{\tabu@ls@G}%
280 \fi
281 \tabu@everyrow@egroup
282 }% \tabulinestyle

```

**`\taburulecolor`** [colortbl](#) provides `\arrayrulecolor`, but the definition is global and must be restored manually after the table. `\taburulecolor` works with the same scheme as `\everyrow`: even if the definition of the rules colors must be global (because we it can be changed inside the tabular) the value is not restored globally at the end of the environment.

Instead, `\tabu@arc@L` stores locally the color definition (*ie.* its definition is relative to the group level before the entry inside the `tabu` environment).

This is the same for `\doublerulesepcolor` (which may be given as an optional argument to `\taburulecolor`): [colortbl](#) makes the definition global, while  $\mathcal{T}_N b c$  keeps grouping level into mind (“locally-global” settings).

```

283 \newcommand*\taburulecolor{\tabu@everyrow@bgroup \tabu@textbar \tabu@rulecolor}
284 \def\tabu@rulecolor #1{\toks@{}}%
285 \def\tabu@temp #1#1#1{\tabu@ruledrsc{##1}}\@ifnextchar #1%
286 \tabu@temp
287 \tabu@rulearc
288 }% \tabu@rulecolor
289 \def\tabu@ruledrsc #1{\edef\tabu@temp{#1}\tabu@strtrim\tabu@temp
290 \ifx \tabu@temp\empty \def\tabu@temp{\tabu@rule@drsc@ {}{}}}%
291 \else \edef\tabu@temp{\noexpand\tabu@rule@drsc@ {}{\tabu@temp}}}%
292 \fi
293 \tabu@temp
294 }% \tabu@ruledrsc@
295 \def\tabu@ruledrsc@ #1#{\tabu@rule@drsc@ {#1}}
296 \def\tabu@rule@drsc@ #1#2{%
297 \iftabu@everyrow
298 \ifx \\\#1#2\\\toks@{\let\CT@drsc@ \relax}%
299 \else \toks@{\def\CT@drsc@{\color #1{#2}}}%
300 \fi
301 \else
302 \ifx \\\#1#2\\\global\let\CT@drsc@ \relax
303 \else \gdef\CT@drsc@{\color #1{#2}}%
304 \fi
305 \fi
306 \tabu@rulearc
307 }% \tabu@rule@drsc@
308 \def\tabu@rulearc #1#{\tabu@rule@arc@ {#1}}
309 \def\tabu@rule@arc@ #1#2{%
310 \iftabu@everyrow
311 \ifx \\\#1#2\\\toks@{\expandafter{\the\toks@ \def\CT@arc@{}}}%
312 \else \toks@{\expandafter{\the\toks@ \def\CT@arc@{\color #1{#2}}}%
313 \fi
314 \toks@{\expandafter{\the\toks@
315 \let\tabu@arc@L \CT@arc@
316 \let\tabu@drsc@L \CT@drsc@}}%
317 \else
318 \ifx \\\#1#2\\\gdef\CT@arc@{}%
319 \else \gdef\CT@arc@{\color #1{#2}}%
320 \fi

```

```

321      \global\let\tabu@arc@G \CT@arc@
322      \global\let\tabu@drsc@G \CT@drsc@
323      \fi
324      \tabu@everyrow@egroup
325 }% \tabu@rule@arc@

```

**\taburowcolors** **\taburowcolors** {number}<number>{first color .. last color}

The aim of the game is to define the process that will be executed at **\everyrow**.

After that, the usual process for “locally-global” settings is plugged into **\tabu@cleanup** and **\tabu@reset...**

```

326 \def\taburowcolors {\tabu@everyrow@bgroup \@testopt \tabu@rowcolors 1}
327 \def\tabu@rowcolors [#1]#2#{\tabu@rowcolor@lors{#1}{#2}}
328 \def\tabu@rowcolor@lors #1#2#3{%
329     \toks@{\@defaultunits \count@      =\number0#2\relax \@nnil
330         \@defaultunits \tabu@start    =\number0#1\relax \@nnil
331     \ifnum \count@<\tw@ \count@=\tw@ \fi
332     \advance\tabu@start \m@ne
333     \ifnum \tabu@start<\z@ \tabu@start \z@ \fi
334     \tabu@rowcolorseries #3\in@..\in@ \@nnil
335 }% \tabu@rowcolors
336 \def\tabu@rowcolorseries #1..#2\in@ #3\@nnil {%
337     \ifx \in@#1\relax
338         \iftabu@everyrow \toks@{\def\tabu@rc@{\let\tabu@rc@L \tabu@rc@}}%
339         \else \gdef\tabu@rc@{\global\let\tabu@rc@G \tabu@rc@}
340         \fi
341     \else
342         \ifx \\#2\\ \tabu@rowcolorserieserror \fi
343         \tabu@sanitizearg{#1}\tabu@temp
344         \tabu@sanitizearg{#2}\@tempa
345         \advance\count@ \m@ne
346     \iftabu@everyrow
347         \def\tabu@rc@ ##1##2##3##4{\def\tabu@rc@{%
348             \ifnum ##2=\c@taburow
349                 \definecolorseries\tabu@rcseries@{the\tabu@nested}{rgb}{last}{##3}{##4}\fi
350             \ifnum \c@taburow<##2 \else
351                 \ifnum \tabu@modulo {\c@taburow-##2}{##1+1}=\z@
352                     \resetcolorseries[{##1}]{tabu@rcseries@{the\tabu@nested}}\fi
353                 \xglobal\colorlet\tabu@rc@{the\tabu@nested}{tabu@rcseries@{the\tabu@nested}!!+}%
354                 \rowcolor\tabu@rc@{the\tabu@nested}\fi}%
355             }\edef\x{\noexpand\tabu@rc@      {\the\count@}
356                                     {\the\tabu@start}
357                                     {\tabu@temp}
358                                     {\@tempa}}%
359             }\x
360         \toks@\expandafter{\expandafter\def\expandafter\tabu@rc@\expandafter{\tabu@rc@}}%
361         \toks@\expandafter{\the\toks@ \let\tabu@rc@L \tabu@rc@}%
362     \else % inside \noalign
363         \definecolorseries\tabu@rcseries@{the\tabu@nested}{rgb}{last}{\tabu@temp}{\@tempa}%
364         \expandafter\resetcolorseries\expandafter[\the\count@]{tabu@rcseries@{the\tabu@nested}}%
365         \xglobal\colorlet\tabu@rc@{the\tabu@nested}{tabu@rcseries@{the\tabu@nested}!!+}%
366         \let\noalign \relax \rowcolor\tabu@rc@{the\tabu@nested}%
367         \def\tabu@rc@ ##1##2{\gdef\tabu@rc@{%
368             \ifnum \tabu@modulo {\c@taburow-##2}{##1+1}=\@ne
369                 \resetcolorseries[{##1}]{tabu@rcseries@{the\tabu@nested}}\fi
370             \xglobal\colorlet\tabu@rc@{the\tabu@nested}{tabu@rcseries@{the\tabu@nested}!!+}%
371             \rowcolor\tabu@rc@{the\tabu@nested}}}%
372     }\edef\x{\noexpand\tabu@rc@{\the\count@}{\the\c@taburow}}\x
373     \global\let\tabu@rc@G \tabu@rc@

```

```

374 \fi
375 \fi
376 \tabu@everyrow@egroup
377 }% \tabu@rowcolorseries
378 \tabuDisableCommands {\let\tabu@rc@ \@empty }
379 \def\tabu@rowcolorserieserror {\PackageError{tabu}
380 {Invalid syntax for \string\taburowcolors
381 \MessageBreak Please look at the documentation!}\@ehd
382 }% \tabu@rowcolorserieserror

```

**\tabureset** Simply – and locally – reset the default values for `\tabulinesep` (0pt), `\extrarowsep` (0pt), `\extratabsurround` (0pt), `\tabulinestyle` {}, `\everyrow` {} and `\taburulecolor` []{}.

```

383 \newcommand*\tabureset {%
384 \tabulinesep=\z@ \extrarowsep=\z@ \extratabsurround=\z@
385 \tabulinestyle{}\everyrow{}\taburulecolor|{} \taburowcolors{}%
386 }% \tabureset

```

## Parsing line styles

**\tabu@getline** This macro parses a line specification argument of the form:

3pt BlannedAlmond on 4pt Crimsom off 2pt ForestGreen

Note that Crimsom will overwrite BlannedAlmond in this case: the color for the line dash may be specified after the line width or after the line dash length.

The process uses `\scantokens` on the argument given by the user, which is first expanded in a context where the `babel` switch `\if@save@actives` is set to `true`. Then `\scantokens` is used on the argument in a group where the letter “o” is active, and defined to be a macro which rewrites the line specification. Incidentally, the comma is active too, and expands to a space. This way the initial argument is “genetically modified”, so that it becomes very easy to assign dimensions (thickness, dash length and gap length) and colors separately.

For example: 3pt BlannedAlmond on 4pt Crimsom will be expanded in a context where “o” is active (and equal to `\tabu@oxiii`, the `xiii` suffix means “active” *ie.* `\catcode = 13`).

Then the “o” in `BlannedAlmond` is rewritten as follow:

1. “o” sees “n” after itself, then it expands `\tabu@onxiii`.
2. `\tabu@onxiii` sees a character whose catcode is not other, then the rewriting process is aborted, and “ond” is rewritten as “ond” where the “o” is not active but the usual letter “o”.

The next “o” is rewritten as follow:

1. “o” sees “n” after itself, then it expands `\tabu@onxiii`.
2. `\tabu@onxiii` sees a space (which is active): it calls back itself again,
3. `\tabu@onxiii` sees a character whose catcode is other: then the sequence “on<sub>L</sub>3” is rewritten as:  
“`\tabu@ \tabu@on =4pt Crimsom`”

Finally the whole argument is rewritten as:

`\tabu@ \tabu@thick =3pt BlannedAlmond \tabu@ \tabu@on =4pt Crimsom \tabu@ \tabu@`

Define `\tabu@` as an appropriate macro which uses `\afterassignment` to:

1. Assign the corresponding dimension (thickness, dash length or gap length).
2. Collect the rest until the next `\tabu@`, trim spaces and check if the color exists.

Limitation: A color name must not contain a sequence that matches on of the patterns:

`...on<a character of category 12>...` or `...off<a character of category 12>...`

But this “limitation” is not too heavy, I suppose...



The result is `\tabu@thestyle`: a `tabu` line style to be used to rewrite a `|` column, for `\tabucline`.

We use locally the L<sup>A</sup>T<sub>E</sub>X defined dimen registers `\@tempdima`, `\@tempdimb` and `\@tempdimc`. For clarity, their names are `\tabu@thick`, `\tabu@on` and `\tabu@off` here...

```

387 \def\tabu@getline #1{\begingroup
388   \csname \ifcsname if@safe@actives\endcsname          % <babel>
389   @safe@activetrue\else
390   relax\fi      \endcsname
391   \edef\tabu@temp{#1}\tabu@sanitizearg{#1}\@tempa
392   \let\tabu@thestyle \relax
393   \ifcsname tabu@linestyle@\@tempa \endcsname
394   \edef\tabu@thestyle{\endgroup
395   \def\tabu@thestyle{\expandafter\noexpand
396   \csname tabu@linestyle@\@tempa\endcsname}%
397   }\tabu@thestyle
398   \else \expandafter\tabu@definestyle \tabu@temp \@nil
399   \fi
400 }% \tabu@getline

```

**`\tabu@definestyle`** Here is the `\scantokens` stuff.

```

401 \def\tabu@definestyle #1#2\@nil {\endlinechar \m@ne \makeatletter
402   \tabu@thick \maxdimen \tabu@on \maxdimen \tabu@off \maxdimen
403   \let\tabu@c@lon \@undefined \let\tabu@c@loff \@undefined
404   \ifodd 1\ifcat .#1\else\ifcat\relax #1\else 0\fi\fi % catcode 12 or non expandable cs
405   \def\tabu@temp{\tabu@getparam{thick}}%
406   \else \def\tabu@temp{\tabu@getparam{thick}\maxdimen}%
407   \fi
408   {%
409     \let\tabu@ \relax
410     \def\:{\obeyspaces \tabu@oXIII \tabu@commaXIII \edef\:% (space active \: happy ;-)}
411     \scantokens{\:{\tabu@temp #1#2 \tabu@\tabu@}}%
412     \expandafter}\expandafter
413     \def\expandafter\:\expandafter{\:% line spec rewritten now ;-)}
414     \def\;{\def\:%}
415     \scantokens\expandafter{\expandafter\;\expandafter{\:%} space is now inactive (catcode 10)}
416     \let\tabu@ \tabu@getcolor \: % all arguments are ready now ;-)}
417     \ifdefined\tabu@c@lon \else \let\tabu@c@lon\@empty \fi
418     \ifx \tabu@c@lon\@empty \def\tabu@c@lon{\CT@arc@}\fi
419     \ifdefined\tabu@c@loff \else \let\tabu@c@loff \@empty \fi
420     \ifdim \tabu@on=\maxdimen \ifdim \tabu@off<\maxdimen
421       \tabu@on \tabulineon \fi\fi
422     \ifdim \tabu@off=\maxdimen \ifdim \tabu@on<\maxdimen
423       \tabu@off \tabulineoff \fi\fi
424     \ifodd 1\ifdim \tabu@off=\maxdimen \ifdim \tabu@on=\maxdimen 0 \fi\fi
425       \in@true % <leaders>
426     \else \in@false % <rule>
427     \fi
428     \ifdim\tabu@thick=\maxdimen \def\tabu@thick{\arrayrulewidth}%
429     \else \edef\tabu@thick{\the\tabu@thick}%
430     \fi
431     \edef \tabu@thestyle ##1##2{\endgroup
432     \def\tabu@thestyle{%
433       \ifin@ \noexpand\tabu@leadersstyle {\tabu@thick}
434       {\the\tabu@on}{##1}
435       {\the\tabu@off}{##2}%
436     \else \noexpand\tabu@rulesstyle
437       {##1\vrule width \tabu@thick}%
438       {##1\leaders \hrule height \tabu@thick \hfil}%

```

```

439         \fi}%
440     }\expandafter \expandafter
441     \expandafter \tabu@thestyle \expandafter
442     \expandafter \expandafter
443     {\expandafter\tabu@c@lon\expandafter}\expandafter{\tabu@c@loff}%
444 }% \tabu@definestyle

```

**\tabu@onxiii** We have to define the active “o” character, which looks for the next tokens, trying to find a pattern like **on**(*category 12*) or **off**(*category 12*) (possibly with – active – spaces between **on** or **off** and the next character of catcode 12).

**\tabu@ofxiii**  
**\tabu@offiii**

```

445 {\catcode`\O=\active \lccode`\O=`\o \catcode`\,=\active
446     \lowercase{\gdef\tabu@oXIII {\catcode`\o=\active \let O=\tabu@oxiii}}
447     \gdef\tabu@commaXIII {\catcode`\,=\active \let ,=\space}
448 }% \catcode
449 \def\tabu@oxiii #1{%
450     \ifcase \ifx n#1\z@ \else
451         \ifx f#1\@ne\else
452             \tw@ \fi\fi
453         \expandafter\tabu@onxiii
454     \or \expandafter\tabu@ofxiii
455     \else o%
456     \fi#1}%
457 \def\tabu@onxiii #1#2{%
458     \ifcase \ifx !#2\tw@ \else
459         \ifcat.\noexpand#2\z@ \else
460             \ifx \tabu@spxiii#2\@ne\else
461                 \tw@ \fi\fi\fi
462             \tabu@getparam{on}#2\expandafter\@gobble
463         \or \expandafter\tabu@onxiii % (space is active)
464         \else o\expandafter\@firstofone
465         \fi{#1#2}}%
466 \def\tabu@ofxiii #1#2{%
467     \ifx #2f\expandafter\tabu@offxiii
468     \else o\expandafter\@firstofone
469     \fi{#1#2}}
470 \def\tabu@offxiii #1#2{%
471     \ifcase \ifx !#2\tw@ \else
472         \ifcat.\noexpand#2\z@ \else
473             \ifx \tabu@spxiii#2\@ne \else
474                 \tw@ \fi\fi\fi
475             \tabu@getparam{off}#2\expandafter\@gobble
476         \or \expandafter\tabu@offxiii % (space is active)
477         \else o\expandafter\@firstofone
478         \fi{#1#2}}

```

**\tabu@getparam** The rewritten stuff.

```

479 \def\tabu@getparam #1{\tabu@ \csname tabu@#1\endcsname=}

```

**\tabu@getcolor** **\tabu@ \tabu@on** = *3pt* **Crimson****\tabu@**

**\tabu@getcolor** first makes the assignment to **\tabu@on** and then looks for the color name which might have been placed before the next **\tabu@**.

```

480 \def\tabu@getcolor #1{% \tabu@ <- \tabu@getcolor after \edef
481     \ifx \tabu@#1\else % no more spec
482         \let\tabu@theparam=#1\afterassignment \tabu@getc@l@r #1\fi
483 }% \tabu@getcolor
484 \def\tabu@getc@l@r #1\tabu@ {%
485     \def\tabu@temp{#1}\tabu@strtrim \tabu@temp

```

```

486 \ifx \tabu@temp\@empty
487 \else%\ifcsname \string\color@\tabu@temp \endcsname % if the color exists
488 \ifx \tabu@theparam \tabu@off \let\tabu@c@loff \tabu@c@l@r
489 \else \let\tabu@c@lon \tabu@c@l@r
490 \fi
491 %\else \tabu@warncolour{\tabu@temp}%
492 \fi%\fi
493 \tabu@ % next spec
494 }% \tabu@getc@l@r
495 \def\tabu@warncolour #1{\PackageWarning\tabu
496 {Color #1 is not defined. Default color used}%
497 }% \tabu@warncolour

```

**\tabu@leadersstyle** When a style is executed, it expands either **\tabu@leadersstyle** or **\tabu@rulesstyle** depending on whether **\tabu@rulesstyle** or not it contains leaders (dashed lines) or simple rules (solid lines): T<sub>E</sub>X internals allow to insert solid lines easily inside a tabular, while inserting leaders is more complex.

**\tabu@leadersstyle** eventually rebuilds the (horizontal and vertical) leaders boxes, and then define two macros: **\tabu@thevleaders** and **\tabu@thehleads**, suitable to draw vertical and horizontal lines respectively. Incidentally, **\tabu@leaders** is defined to be the parameters for the leaders.

**\tabu@rulesstyle** only defines the two macros **\tabu@thevrule** and **\tabu@thehrule**. The control sequence **\tabu@leaders** is undefined so that we know if the style contains a leader or a rule.

```

498 \def\tabu@leadersstyle #1#2#3#4#5{\def\tabu@leaders{{#1}{#2}{#3}{#4}{#5}}%
499 \ifx \tabu@leaders\tabu@leaders@G \else
500 \tabu@LEADERS{#1}{#2}{#3}{#4}{#5}\fi
501 }% \tabu@leadersstyle
502 \def\tabu@rulesstyle #1#2{\let\tabu@leaders \@undefined
503 \gdef\tabu@thevrule{#1}\gdef\tabu@thehrule{#2}%
504 }% \tabu@rulesstyle

```

**\tabu@LEADERS** Here the two leaders boxes **\tabu@hleads** and **\tabu@vleads** are built, as well as the leaders macros **\tabu@thehleaders** and **\tabu@thevleaders**.

```

505 \def\tabu@LEADERS #1#2#3#4#5{%% width, dash, dash color, gap, gap color
506 {\let\color \tabu@color % => during trials -> \color = \tabu@nocolor
507 % % but the leaders boxes should have colors !
508 \def\@therule{\vrule}\def\@thick{height}\def\@length{width}%
509 \def\@box{\hbox}\def\@unbox{\unhbox}\def\@elt{\wd}%
510 \def\@skip{\hskip}\def\@ss{\hss}\def\tabu@leads{\tabu@hleads}%
511 \tabu@l@ed@rs {#1}{#2}{#3}{#4}{#5}%
512 \global\let\tabu@thehleaders \tabu@theleaders
513 }%
514 {%
515 \def\@therule{\hrule}\def\@thick{width}\def\@length{height}%
516 \def\@box{\vbox}\def\@unbox{\unvbox}\def\@elt{\ht}%
517 \def\@skip{\vskip}\def\@ss{\vss}\def\tabu@leads{\tabu@vleads}%
518 \tabu@l@ed@rs {#1}{#2}{#3}{#4}{#5}%
519 \global\let\tabu@thevleaders \tabu@theleaders
520 }%
521 \gdef\tabu@leaders@G{{#1}{#2}{#3}{#4}{#5}}%
522 }%
523 }% \tabu@LEADERS
524 \def\tabu@therule #1#2{\@therule \@thick#1\@length\dimexpr#2/2 \@depth\z@}
525 \def\tabu@l@ed@rs #1#2#3#4#5{%% width, dash, dash color, gap, gap color
526 \global\setbox \tabu@leads=\@box{%
527 {#3\tabu@therule{#1}{#2}}%
528 \ifx\\#5\\ \@skip#4\else{#5\tabu@therule{#1}{#4*2}}\fi
529 {#3\tabu@therule{#1}{#2}}}%
530 \global\setbox\tabu@leads=\@box to\@elt\tabu@leads{\@ss

```

```

531      {#3\tabu@therule{#1}{#2}}\@unbox\tabu@leads}%
532      \edef\tabu@theleaders ##1{\def\noexpand\tabu@theleaders {%
533      {##1\tabu@therule{#1}{#2}}}%
534      \xleaders \copy\tabu@leads \@ss
535      \tabu@therule{0pt}{-#2}{##1\tabu@therule{#1}{#2}}}%
536      }\tabu@theleaders{#3}%
537 }% \tabu@l@@@ders

```

## 11.8 The entry inside tabu

### `\tabu`, `\endtabu`, `\longtabu` and `\endlontabu`

`\tabu` `\tabu` and `\longtabu` are the commands of the environments.

`\endtabu` `\endtabu` is `\endtabular` or `\endarray` in math mode.

```

538 \newcommand*\tabu {\tabu@longfalse
539      \ifmmode \def\tabu@ {\array}\def\endtabu {\endarray}%
540      \else \def\tabu@ {\tabu@tabular}\def\endtabu {\endtabular}\fi
541      \expandafter\let\csname tabu*\endcsname \tabu
542      \expandafter\def\csname endtabu*\endcsname{\endtabu}%
543      \tabu@spreadfalse \tabu@negcoeffalse \tabu@settarget
544 }% {tabu}
545 \let\tabu@tabular \tabular % <For LyX: some users redefine \tabular...>
546 \expandafter\def\csname tabu*\endcsname{\tabuscantokenstrue \tabu}
547 \newcommand*\longtabu {\tabu@longtrue
548      \ifmmode\PackageError{tabu}{longtabu not allowed in math mode}\fi
549      \def\tabu@{\longtable}\def\endlontabu{\endlongtable}%
550      \LTchunksiz=\@M
551      \expandafter\let\csname tabu*\endcsname\tabu
552      \let\LT@startpbox \tabu\LT@startpbox % \everypar{ array struts }
553      \tabu@spreadfalse \tabu@negcoeffalse \tabu@settarget
554 }% {longtabu}
555 \expandafter\def\csname longtabu*\endcsname{\tabuscantokenstrue \longtabu}
556 \def\tabu@nolongtabu{\PackageError{tabu}
557      {longtabu requires the longtable package}\@ehd}

```

### Setting the tabu target

`\tabu@settarget` The macro sets `\tabu@target` (a dimen) to the value specified for “tabu to” or “tabu spread”.

```

\tabu@begin
558 \def\tabu@settarget {\futurelet\@let@token \tabu@sett@rget }
559 \def\tabu@sett@rget {\tabu@target \z@
560      \ifcase \ifx \bgroup\@let@token \z@ \else
561      \ifx \@sptoken\@let@token \@ne \else
562      \if t\@let@token \tw@ \else
563      \if s\@let@token \thr@@\else
564      \z@\fi\fi\fi\fi
565      \expandafter\tabu@begin
566      \or \expandafter\tabu@gobblespace\expandafter\tabu@settarget
567      \or \expandafter\tabu@to
568      \or \expandafter\tabu@spread
569      \fi
570 }% \tabu@sett@rget
571 \def\tabu@to to{\def\tabu@halignto{to}\tabu@gettarget}
572 \def\tabu@spread spread{\def\tabu@spreadtrue\def\tabu@halignto{spread}\tabu@gettarget}
573 \def\tabu@gettarget {\afterassignment\tabu@linegoaltarget \tabu@target }
574 \def\tabu@linegoaltarget {\futurelet\tabu@temp \tabu@linegoalt@rget }
575 \def\tabu@linegoalt@rget {%
576      \ifx \tabu@temp\LNG\@setlinegoal

```

```

577 \LNGl@setlinegoal \expandafter \@firstoftwo \fi % @gobbles \LNGl@setlinegoal
578 \tabu@begin
579 }% \tabu@linegoal\@target
580 \def\tabu@begin #1#{%
581 \iftabu@measuring \expandafter\tabu@nestedmeasure \fi
582 \ifdim \tabu@target=\z@ \let\tabu@halignto \@empty
583 \else \edef\tabu@halignto{\tabu@halignto\the\tabu@target}%
584 \fi
585 \@testopt \tabu@tabu@ \tabu@aligndefault #1\@nil
586 }% \tabu@begin
587 \long\def\tabu@tabu@ [#1]#2\@nil #3{\tabu@setup
588 \def\tabu@align {#1}\def\tabu@savdpream{\NC@find #3}%
589 \tabu@ [\tabu@align ]#2{#3\tabu@rewritefirst }%
590 }% \tabu@tabu@
591 \def\tabu@nestedmeasure {%
592 \ifodd 1\iftabu@spread \else \ifdim\tabu@target=\z@ \else 0 \fi\fi\relax
593 \tabu@spreadtrue
594 \else \beginingroup \iffalse{\fi \ifnum0=}\fi
595 \toks@{}\def\tabu@stack{b}%
596 \expandafter\tabu@collectbody\expandafter\tabu@quickrule
597 \expandafter\endgroup
598 \fi
599 }% \tabu@nestedmeasure
600 \def\tabu@quickrule {\indent\vrule height\z@ depth\z@ width\tabu@target}

```

**\tabu@setup** **\tabu@init** is expanded only when **tabu** is not nested. In this case, and if **\parindent > 0**, and if **\tabu@init** **\tabu@defaulttarget = \linewidth**, the correction of the default target for paragraph indentation is executed (see [paragraph indentation](#)).

**\tabu@indent**

```

601 \def\tabu@setup{\tabu@alloc@
602 \ifcase \tabu@nested
603 \ifmmode \else \ifdim\tabu@target=\z@ \let\tabu@afterendpar \par \fi\fi
604 \def\tabu@aligndefault{c}\tabu@init \tabu@indent
605 \else % <nested tabu>
606 \def\tabu@aligndefault{t}\let\tabu@defaulttarget \linewidth
607 \fi
608 \let\tabu@thetarget \tabu@defaulttarget \let\tabu@restored \@undefined
609 \edef\tabu@NC@list{\the\NC@list}\NC@list{\NC@do \tabu@rewritefirst}%
610 \everycr{}\let\@startpbox \tabu@startpbox % for nested tabu inside longtabu...
611 \let\@endpbox \tabu@endpbox % idem " " " " " "
612 \let\@tabarray \tabu@tabarray % idem " " " " " "
613 \tabu@setcleanup \tabu@setreset
614 }% \tabu@setup
615 \def\tabu@init{\tabu@starttimer \tabu@measuringfalse
616 \edef\tabu@hfuzz {\the\dimexpr\hfuzz+1sp}\global\tabu@footnotes{}%
617 \let\firstline \tabu@firstline \let\lastline \tabu@lastline
618 \let\firstline \tabu@firstline \let\lastline \tabu@lastline
619 \let\hline \tabu@hline \let\@xhline \tabu@xhline
620 \let\color \tabu@color \let\@arstrutbox \tabu@arstrutbox
621 \tabu@trivlist %<restore \\=\@normalcr inside lists>
622 \let\@footnotetext \tabu@footnotetext \let\@xfootnotetext \tabu@xfootnotetext
623 \let\@xfootnote \tabu@xfootnote \let\centering \tabu@centering
624 \let\raggedright \tabu@raggedright \let\raggedleft \tabu@raggedleft
625 \let\tabudecimal \tabu@tabudecimal \let\Centering \tabu@Centering
626 \let\RaggedRight \tabu@RaggedRight \let\RaggedLeft \tabu@RaggedLeft
627 \let\justifying \tabu@justifying \let\rowfont \tabu@rowfont
628 \let\fbbox \tabu@fbbox \let\color@b@x \tabu@color@b@x
629 \let\tabu@@everycr \everycr \let\tabu@@everypar \everypar
630 \let\tabu@prepnext@tokORI \prepnext@tok\let\prepnext@tok \tabu@prepnext@tok

```

```

631 \let\tabu@multicolumnORI\multicolumn \let\multicolumn \tabu@multicolumn
632 \let\tabu@startpbox \@startpbox % for nested tabu inside longtabu pfff !!!
633 \let\tabu@endpbox \@endpbox % idem " " " " " " " "
634 \let\tabu@tabarray \@tabarray % idem " " " " " " " "
635 \tabu@adl@fix \let\endarray \tabu@endarray % <fix> colortbl & arydshln (delarray)
636 \iftabu@colortbl\CT@everycr\expandafter{\expandafter\iftabu@everyrow \the\CT@everycr \fi}\fi
637 }% \tabu@init
638 \def\tabu@indent{% correction for indentation
639 \ifdim \parindent>\z@\ifx \linewidth\tabubefaulttarget
640 %
641 \everypar\expandafter{%
642 \the\everypar\everypar\expandafter{\the\everypar}%
643 \setbox\z@=\lastbox
644 \ifdim\wd\z@>\z@ \edef\tabu@thetarget
645 {\the\dimexpr -\wd\z@+\tabubefaulttarget}\fi
646 \box\z@}%
647 \fi\fi
648 }% \tabu@indent

```

**\tabu@setcleanup** We have to save locally (in the group of the environment) the current value of the last global assignments to \CT@arc@, \CT@drsc@, \tabu@ls@ etc.

**\tabu@cleanup** Restoration will be done globally after the box that contains the tabular by \tabu@cleanup.

```

649 \def\tabu@setcleanup {% saves last global assignments
650 \ifodd 1\ifmmode \else \iftabu@long \else 0\fi\fi\relax
651 \def\tabu@aftergroupcleanup{%
652 \def\tabu@aftergroupcleanup{\aftergroup\tabu@cleanup}}%
653 \else
654 \def\tabu@aftergroupcleanup{%
655 \aftergroup\aftergroup\aftergroup\tabu@cleanup
656 \let\tabu@aftergroupcleanup \relax}%
657 \fi
658 \let\tabu@arc@Gsave \tabu@arc@G
659 \let\tabu@arc@G \tabu@arc@L % <init>
660 \let\tabu@drsc@Gsave \tabu@drsc@G
661 \let\tabu@drsc@G \tabu@drsc@L % <init>
662 \let\tabu@ls@Gsave \tabu@ls@G
663 \let\tabu@ls@G \tabu@ls@L % <init>
664 \let\tabu@rc@Gsave \tabu@rc@G
665 \let\tabu@rc@G \tabu@rc@L % <init>
666 \let\tabu@evr@Gsave \tabu@evr@G
667 \let\tabu@evr@G \tabu@evr@L % <init>
668 \let\tabu@celllalign@save \tabu@celllalign
669 \let\tabu@celllralign@save \tabu@celllralign
670 \let\tabu@cellleft@save \tabu@cellleft
671 \let\tabu@cellright@save \tabu@cellright
672 \let\tabu@@celllalign@save \tabu@@celllalign
673 \let\tabu@@celllralign@save \tabu@@celllralign
674 \let\tabu@@cellleft@save \tabu@@cellleft
675 \let\tabu@@cellright@save \tabu@@cellright
676 \let\tabu@rowfontreset@save \tabu@rowfontreset
677 \let\tabu@@rowfontreset@save\tabu@@rowfontreset
678 \let\tabu@rowfontreset \@empty
679 \edef\tabu@alloc@save {\the\tabu@alloc}% restore at \tabu@reset
680 \edef\c@taburow@save {\the\c@taburow}%
681 \edef\tabu@naturalX@save {\the\tabu@naturalX}%
682 \let\tabu@naturalXmin@save \tabu@naturalXmin
683 \let\tabu@naturalXmax@save \tabu@naturalXmax

```

```

684 \let\tabu@mkarstrut@save \tabu@mkarstrut
685 \edef\tabu@clarstrut{%
686     \extrarowheight \the\dimexpr \ht\@arstrutbox-\ht\strutbox \relax
687     \extrarowdepth \the\dimexpr \dp\@arstrutbox-\dp\strutbox \relax
688     \let\noexpand\@arraystretch \@ne \noexpand\tabu@rearstrut}%
689 }% \tabu@setcleanup
690 \def\tabu@cleanup {\begingroup
691     \globaldefs\@ne \tabu@everyrowtrue
692     \let\tabu@arc@G \tabu@arc@Gsave
693     \let\CT@arc@ \tabu@arc@G
694     \let\tabu@drsc@G \tabu@drsc@Gsave
695     \let\CT@drsc@ \tabu@drsc@G
696     \let\tabu@ls@G \tabu@ls@Gsave
697     \let\tabu@ls@ \tabu@ls@G
698     \let\tabu@rc@G \tabu@rc@Gsave
699     \let\tabu@rc@ \tabu@rc@G
700     \let\CT@do@color \relax
701     \let\tabu@evr@G \tabu@evr@Gsave
702     \let\tabu@celllalign \tabu@celllalign@save
703     \let\tabu@cellralign \tabu@cellralign@save
704     \let\tabu@cellleft \tabu@cellleft@save
705     \let\tabu@cellright \tabu@cellright@save
706     \let\tabu@@celllalign \tabu@@celllalign@save
707     \let\tabu@@cellralign \tabu@@cellralign@save
708     \let\tabu@@cellleft \tabu@@cellleft@save
709     \let\tabu@@cellright \tabu@@cellright@save
710     \let\tabu@rowfontreset \tabu@rowfontreset@save
711     \let\tabu@@rowfontreset \tabu@@rowfontreset@save
712     \tabu@naturalX =\tabu@naturalX@save
713     \let\tabu@naturalXmax \tabu@naturalXmax@save
714     \let\tabu@naturalXmin \tabu@naturalXmin@save
715     \let\tabu@mkarstrut \tabu@mkarstrut@save
716     \c@taburow =\c@taburow@save
717     \ifcase \tabu@nested \tabu@alloc \m@ne\fi
718     \endgroup % <end of \globaldefs>
719     \ifcase \tabu@nested
720         \the\tabu@footnotes \global\tabu@footnotes{}%
721         \tabu@afterendpar \tabu@elapsedtime
722     \fi
723     \tabu@clarstrut
724     \everyrow\expandafter {\tabu@evr@G}%
725 }% \tabu@cleanup
726 \let\tabu@afterendpar \relax

```

**\tabu@setreset** At the beginning of each trial, we have to restore the current value that were active at the entry in the **tabu** environment (for they could have been globally overwritten inside the tabular).

The same must occur when using **\usetabu** as a preamble. Values are restored locally inside the **tabu** box.

**\tabu@reset** **\tabu@setreset** defines **\tabu@reset** to be expanded at the beginning of each trial and when **\usetabu** is used.

```

727 \def\tabu@setreset {%
728     \edef\tabu@savparams {% \relax for \tabu@message@save
729         \ifmmode \col@sep \the\arraycolsep
730         \else \col@sep \the\tabcolsep \fi \relax
731         \arrayrulewidth \the\arrayrulewidth \relax
732         \doublerulesep \the\doublerulesep \relax
733         \extratabsurround \the\extratabsurround \relax
734         \extrarowheight \the\extrarowheight \relax

```



```

735 \extrarowdepth \the\extrarowdepth \relax
736 \def\noexpand\arraystretch{\arraystretch}%
737 \ifdefined\minrowclearance \minrowclearance\the\minrowclearance\relax\fi}%
738 \begingroup
739 \@temptokena\expandafter{\tabu@sav$params}% => only for \savetabu / \usetabu
740 \ifx \tabu@arc@L\relax \else \tabu@setsave \tabu@arc@L \fi
741 \ifx \tabu@drsc@L\relax \else \tabu@setsave \tabu@drsc@L \fi
742 \tabu@setsave \tabu@ls@L \tabu@setsave \tabu@evr@L
743 \expandafter \endgroup \expandafter
744 \def\expandafter\tabu@sav@ \expandafter{\the\@temptokena
745 \let\tabu@arc@G \tabu@arc@L
746 \let\tabu@drsc@G \tabu@drsc@L
747 \let\tabu@ls@G \tabu@ls@L
748 \let\tabu@rc@G \tabu@rc@L
749 \let\tabu@evr@G \tabu@evr@L}%
750 \def\tabu@reset{\tabu@sav$params
751 \tabu@everyrowtrue \c@taburow \z@
752 \let\CT@arc@ \tabu@arc@L
753 \let\CT@drsc@ \tabu@drsc@L
754 \let\tabu@ls@ \tabu@ls@L
755 \let\tabu@rc@ \tabu@rc@L
756 \global\tabu@alloc \tabu@alloc@save
757 \everyrow\expandafter{\tabu@evr@L}}%
758 }% \tabu@reset
759 \def\tabu@setsave #1{\expandafter\tabu@sets@ve #1\@nil{#1}}
760 \long\def\tabu@sets@ve #1\@nil #2{\@temptokena\expandafter{\the\@temptokena \def#2{#1}}}

```

## 11.9 The rewriting process: inside the “\@mkpream group”

### New column types and private (new) column types

**\tabu@newcolumnntype** A helper macro to create new column types for tabu.

The column types **are not appended** to \NC@list in order to keep them local to tabu.

```

761 \def\tabu@newcolumnntype #1{%
762 \expandafter\tabu@new@columntype
763 \csname NC@find@\string#1\expandafter\endcsname
764 \csname NC@rewrite@\string#1\endcsname
765 {#1}%
766 }% \tabu@newcolumnntype
767 \def\tabu@new@columntype #1#2#3{%
768 \def#1##1#3{\NC@{##1}}%
769 \let#2\relax \newcommand*#2%
770 }% \tabu@new@columntype

```

**\tabu@privatecolumnntype** Columns types defined with \tabu@privatecolumnntype are "mounted" only inside the \@mkpream group of tabu.

```

771 \def\tabu@privatecolumnntype #1{%
772 \expandafter\tabu@private@columntype
773 \csname NC@find@\string#1\expandafter\endcsname
774 \csname NC@rewrite@\string#1\expandafter\endcsname
775 \csname tabu@NC@find@\string#1\expandafter\endcsname
776 \csname tabu@NC@rewrite@\string#1\endcsname
777 {#1}%
778 }% \tabu@privatecolumnntype
779 \def\tabu@private@columntype#1#2#3#4{%
780 \g@addto@macro\tabu@privatecolumns{\let#1#3\let#2#4}%
781 \tabu@new@columntype#3#4%
782 }% \tabu@private@columntype

```

```
783 \let\tabu@privatecolumns \@empty
```

## High priority columns

**\tabucolumn** \tabucolumn puts a user-defined column in high priority in the tabu rewriting process.

```
784 \newcommand*\tabucolumn [1]{\expandafter \def \expandafter
785   \tabu@highprioritycolumns\expandafter{\tabu@highprioritycolumns
786   \NC@do #1}}%
787 \let\tabu@highprioritycolumns \@empty
```

## Rewriting vertical lines and leaders

**| (private column type)** This is the rewrite macro for the | column type inside tabu and longtabu.

Vertical lines are *simply rewritten* as special ! columns.

```
788 \tabu@privatecolumnstype |\tabu@rewritevline}
789 \newcommand*\tabu@rewritevline[1][\]{\tabu@vlinearg{#1}%
790   \expandafter \NC@find \tabu@rewritten}
```

**\tabu@lines** The | token for vertical lines may have a special catcode. array.sty makes the test with \if and therefore, it is catcode insensitiv. Here, we use \scantokens and check if | is not an *other* character.

```
791 \def\tabu@lines #1{%
792   \ifx|#1\else \tabu@privatecolumnstype #1{\tabu@rewritevline}\fi
793   \NC@list\expandafter{\the\NC@list \NC@do #1}%
794 }% \tabu@lines@
```

**\tabu@vlinearg** The macro that parses the optional argument of | vertical lines...

```
795 \def\tabu@vlinearg #1{%
796   \ifx\|#1\def\tabu@thestyle {\tabu@ls@}%
797   \else\tabu@getline {#1}%
798   \fi
799   \def\tabu@rewritten ##1{\def\tabu@rewritten{!{##1\tabu@thevline}}}%
800   \expandafter\tabu@rewritten\expandafter{\tabu@thestyle}%
801   \expandafter \tabu@keeppls \tabu@thestyle \@nil
802 }% \tabu@vlinearg
803 \def\tabu@keeppls #1\@nil{%
804   \ifcat $\@cdr #1\@nil $\%
805   \ifx \relax#1\else
806   \ifx \tabu@ls@#1\else
807     \let#1\relax
808     \xdef\tabu@mkpreamblebuffer{\tabu@mkpreamblebuffer
809       \tabu@savels\noexpand#1}\fi\fi\fi
810 }% \tabu@keeppls
811 \def\tabu@thevline {\begingroup
812   \ifdefined\tabu@leaders
813     \setbox\@tempboxa=\vtop to\dimexpr
814       \ht\@arstrutbox+\dp\@arstrutbox{{\tabu@thevleaders}}}%
815     \ht\@tempboxa=\ht\@arstrutbox \dp\@tempboxa=\dp\@arstrutbox
816     \box\@tempboxa
817   \else
818     \tabu@thevrule
819   \fi \endgroup
820 }% \tabu@thevline
821 \def\tabu@savels #1{%
822   \expandafter\let\csname\string#1\endcsname #1%
823   \expandafter\def\expandafter\tabu@reset\expandafter{\tabu@reset
824     \tabu@resetls#1}}%
825 \def\tabu@resetls #1{\expandafter\let\expandafter#1\csname\string#1\endcsname}%
```

## Vertical lines and leaders in the `\multicolumn` preamble

`\tabu@rewritemulticolumn` A special rewrite to allow [...] in `\multicolumn` preamble inside `tabu` environment.

As long as `\multicolumn` begins with `\omit` (via `\multispan`) special care has to be taken: everything shall be purely expandable until `\omit`.

`\multicolumn` is not an environment: no group is opened apart the `\@mkpream` group. We open a semi simple group for `\multicolumn` when inside `tabu`, in order for the setup to be local (in case a user would try to embed a `tabular` inside the argument of `\multicolumn...`)

```

826 \tabu@newcolumnntype \tabu@rewritemulticolumn{%
827   \aftergroup \tabu@endrewritemulticolumn % after \@mkpream group
828   \NC@list{\NC@do *}\tabu@textbar \tabu@lines
829   \tabu@savdecl
830   \tabu@privatecolumns
831   \NC@list\expandafter{\the\expandafter\NC@list \tabu\NC@list}%
832   \let\tabu@savels \relax
833   \NC@find
834 }% \tabu@rewritemulticolumn
835 \def\tabu@endrewritemulticolumn{\gdef\tabu@mkpreambuffer{}\endgroup}
836 \def\tabu@multicolumn{\tabu@ifenvir \tabu@multicolumn \tabu@multicolumnORI}
837 \long\def\tabu@multicolumn #1#2#3{\multispan{#1}\begingroup
838   \tabu@everyrowtrue
839   \NC@list{\NC@do \tabu@rewritemulticolumn}%
840   \expandafter\@gobbletwo % gobbles \multispan{#1}
841   \tabu@multicolumnORI{#1}{\tabu@rewritemulticolumn #2}%
842   {\iftabuscantokens \tabu@rescan \else \expandafter\@firstofone \fi
843    {#3}}%
844 }% \tabu@multicolumn

```

## Rewriting `tabu X` columns

**X (private column type)** This is the rewrite macro for `tabu X` columns. Such a column has an optional argument: the width coefficient for the `tabu X` column whose default value is 1, and may be some alignments parameters. The coefficient is used in the expression: `p{\dimexpr <coef>\tabucolX }`

```

845 \tabu@privatecolumnntype X[1][\begingroup \tabu@siunitx{\endgroup \tabu@rewriteX {#1}}{
846 \def\tabu@nosiunitx #1{#1}{\expandafter \NC@find \tabu@rewritten }
847 \def\tabu@siunitx #1{\@ifnextchar \bgroup
848   {\tabu@rewriteX@Ss{#1}}
849   {\tabu@nosiunitx{#1}}}
850 \def\tabu@rewriteX@Ss #1#2{\@temptokena{}}%
851   \@defaultunits \let\tabu@temp =#2\relax\@nnil
852   \ifodd 1\ifx S\tabu@temp \else \ifx s\tabu@temp \else 0 \fi\fi
853   \def\NC@find{\def\NC@find >####1####2<####3\relax{#1 {####1}{####3}%
854     }\expandafter\NC@find \the\@temptokena \relax
855     }\expandafter\NC@rewrite@S \@gobble #2\relax
856   \else \tabu@siunitxerror
857   \fi
858   \expandafter \NC@find \tabu@rewritten
859 }% \tabu@rewriteX@Ss
860 \def\tabu@siunitxerror {\PackageError{tabu}{Not a S nor s column !
861   \MessageBreak X column can only embed siunitx S or s columns}\@ehd
862 }% \tabu@siunitxerror

```

`\tabu@rewriteX` This macro is expanded by during the rewriting process in case a `X` column is found.

`\tabu@Xsum` (a `dimen`) stores the sum of the (absolute) width coefficients.

For the first `X` column found in the preamble, a special setup occurs:

- if the default target is used (no target specified or `tabu spread` with `X` columns), the target: `\tabu@target`

is set to the default, with a message in the .log file.

- \@halignto is \let to \relax to avoid its expansion in \xdef \@preamble just after \@mkpream. Indeed as long as we have to measure the natural width of the tabular, \@halign must be empty for trial steps.
- The rest of the setup is made \aftergroup (ie. after \xdef \@preamble which occurs inside a group) by \tabu@prep@TRIAL.

```

863 \def\tabu@rewriteX #1#2#3{\tabu@Xarg {#1}{#2}{#3}%
864   \iftabu@measuring
865   \else \tabu@measuringtrue % first X column found in the preamble
866     \let\@halignto \relax   \let\tabu@halignto \relax
867     \iftabu@spread \tabu@spreadtarget \tabu@target \tabu@target \z@
868     \else          \tabu@spreadtarget \z@ \fi
869     \ifdim \tabu@target=\z@
870       \setlength\tabu@target \tabu@thetarget
871       \tabu@message{\tabu@message@defaulttarget}%
872     \else \tabu@message{\tabu@message@target}\fi
873   \fi
874 }% \tabu@rewriteX

```

**\tabu@rewriteXrestore** This macro replaces \tabu@rewriteX in the case of \usetabu.

```

875 \def\tabu@rewriteXrestore #1#2#3{\let\@halignto \relax
876   \def\tabu@rewritten{1}}

```

**\tabu@Xarg** A tedious (and fastidious) macro to parse the optional argument of X columns. The aim is to build \tabu@rewritten which expands to the column specification:

>{alignment} p or m or b {\dimexpr coef \tabucolX \relax }

After that array.sty make it easy: \expandafter \NC@find \tabu@rewritten

```

877 \def\tabu@Xarg #1#2#3{%
878   \advance\tabu@Xcol \@ne   \let\tabu@Xlcr \@empty
879   \let\tabu@Xdisp \@empty   \let\tabu@Xmath \@empty
880   \ifx\@#1\% <shortcut when no option>
881     \def\tabu@rewritten{p}\tabucolX \p@ % <default coef = 1>
882   \else
883     \let\tabu@rewritten \@empty \let\tabu@temp \@empty \tabucolX \z@
884     \tabu@Xparse {}#1\relax
885   \fi
886   \tabu@Xrewritten{#2}{#3}%
887 }% \tabu@Xarg
888 \def\tabu@Xparse #1{\futurelet\@let@token \tabu@Xtest}
889 \expandafter\def\expandafter\tabu@Xparsespace\space{\tabu@Xparse{}}
890 \def\tabu@Xtest{%
891   \ifcase \ifx \relax\@let@token \z@ \else
892     \if ,\@let@token \m@ne\else
893     \if p\@let@token 1\else
894     \if m\@let@token 2\else
895     \if b\@let@token 3\else
896     \if l\@let@token 4\else
897     \if c\@let@token 5\else
898     \if r\@let@token 6\else
899     \if j\@let@token 7\else
900     \if L\@let@token 8\else
901     \if C\@let@token 9\else
902     \if R\@let@token 10\else
903     \if J\@let@token 11\else
904     \ifx \@sptoken\@let@token 12\else
905     \if .\@let@token 13\else

```

```

906         \if -\@let@token 13\else
907         \ifcat $\@let@token 14\else
908         15\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
909     \or \tabu@Xtype {p}%
910     \or \tabu@Xtype {m}%
911     \or \tabu@Xtype {b}%
912     \or \tabu@Xalign \raggedright\relax
913     \or \tabu@Xalign \centering\relax
914     \or \tabu@Xalign \raggedleft\relax
915     \or \tabu@Xalign \tabu@justify\relax
916     \or \tabu@Xalign \RaggedRight\raggedright
917     \or \tabu@Xalign \Centering\centering
918     \or \tabu@Xalign \RaggedLeft\raggedleft
919     \or \tabu@Xalign \justifying\tabu@justify
920     \or \expandafter \tabu@Xparespace
921     \or \expandafter \tabu@Xcoef
922     \or \expandafter \tabu@Xm@th
923     \or \tabu@Xcoef{}%
924     \else\expandafter \tabu@Xparse
925     \fi
926 }% \tabu@Xtest
927 \def\tabu@Xalign #1#2{%
928     \ifx \tabu@Xlcr\@empty \else \PackageWarning{tabu}
929         {Duplicate horizontal alignment specification}\fi
930     \ifdefined#1\def\tabu@Xlcr{#1}\let#1\relax
931     \else \def\tabu@Xlcr{#2}\let#2\relax\fi
932     \expandafter\tabu@Xparse
933 }% \tabu@Xalign
934 \def\tabu@Xtype #1{%
935     \ifx \tabu@rewritten\@empty \else \PackageWarning{tabu}
936         {Duplicate vertical alignment specification}\fi
937     \def\tabu@rewritten{#1}\expandafter\tabu@Xparse
938 }% \tabu@Xtype
939 \def\tabu@Xcoef#1{\edef\tabu@temp{\tabu@temp#1}%
940     \afterassignment\tabu@Xc@ef \tabu@cnt\number\if-#10\fi
941 }% \tabu@Xcoef
942 \def\tabu@Xc@ef{\advance\tabucolX \tabu@temp\the\tabu@cnt\p@
943     \tabu@Xparse{}%
944 }% \tabu@Xc@ef
945 \def\tabu@Xm@th #1{\futurelet \@let@token \tabu@Xd@sp}
946 \def\tabu@Xd@sp{\let\tabu@Xmath=$%
947     \ifx $\@let@token \def\tabu@Xdisp{\displaystyle}%
948     \expandafter\tabu@Xparse
949     \else \expandafter\tabu@Xparse\expandafter{\expandafter}%
950     \fi
951 }% \tabu@Xd@sp

```

**\tabu@Xrewritten** Final step: the whole optional argument has been read, then builds the rewritten column specification.

```

952 \def\tabu@Xrewritten {%
953     \ifx \tabu@rewritten\@empty \def\tabu@rewritten{p}\fi
954     \ifdim \tabucolX<\z@ \tabu@negcoeftrue
955     \else\ifdim \tabucolX=\z@ \tabucolX \p@
956     \fi\fi
957     \edef\tabu@temp{\the\tabu@Xcol}\tabu@stript\tabucolX}%
958     \edef\tabu@Xcoefs{\tabu@Xcoefs \tabu@ \tabu@temp}%
959     \edef\tabu@rewritten ##1##2{\def\noexpand\tabu@rewritten{
960         >\tabu@Xlcr \ifx$\tabu@Xmath$\tabu@Xdisp\fi ##1}%
961         \tabu@rewritten {\tabu@hsize \tabu@temp}%

```

```

962      <{##2\ifx$\tabu@Xmath$\fi}}%
963    }\tabu@rewritten
964 }% \tabu@Xrewritten

```

**\tabu@hsize** **\tabu@hsize {X column number}{X column width coefficient}**

Depending on the sign of the coefficient, and of the stored value for the natural width of the column the X cell belongs to, **\tabu@hsize** returns the wanted width for the *par-box* that contains the cell content.

```

965 \def\tabu@hsize #1#2{%
966   \ifdim #2\p@<\z@
967     \ifdim \tabu@colX=\maxdimen \tabu@wd{#1}\else
968       \ifdim \tabu@wd{#1}<-\#2\tabu@colX \tabu@wd{#1}\else -\#2\tabu@colX\fi
969     \fi
970   \else #2\tabu@colX
971   \fi
972 }% \tabu@hsize

```

## Rewritting **\usetabu** and **\preamble**

The rewritting process is very simple, when all the job has been done cleverly at the time of **\savetabu**!

The **\savetabu** macro is a bit more complex...

**\usetabu (private column type)** **\usetabu** is defined as a **tabu** new column type: loaded only inside the **\@mkpream** group inside the **tabu** environment.

```

973 \tabu@privatecolumnstype \usetabu [1]{%
974   \ifx\\#1\\\tabu@saveerr{}\else
975     \@ifundefined{tabu@saved@\string#1}
976       {\tabu@saveerr{#1}}
977       {\let\tabu@rewriteX \tabu@rewriteXrestore
978        \csname tabu@saved@\string#1\expandafter\endcsname\expandafter\@ne}%
979     \fi
980 }% \NC@rewrite@\usetabu

```

**\preamble (private column type)** **\preamble** is defined as a **tabu** new column type: loaded only inside the **\@mkpream** group inside the **tabu** environment.

```

981 \tabu@privatecolumnstype \preamble [1]{%
982   \ifx\\#1\\\tabu@saveerr{}\else
983     \@ifundefined{tabu@saved@\string#1}
984       {\tabu@saveerr{#1}}
985       {\csname tabu@saved@\string#1\expandafter\endcsname\expandafter\z@}%
986     \fi
987 }% \NC@rewrite@\preamble

```

## Controlling the rewritting process

**\tabu@rewritefirst** This new column type is not really a column type! It is always added to a **tabu** preamble in order to do some setup before any other column is rewritten by **\@mkpream**.

Thus, **\NC@list** is simply set to **{\NC@do \tabu@rewritefirst }**. The rewritting of **\tabu@rewritefirst** will restore the original list **\NC@list**.

This “column type” sets:

- **\tabu@select** to be expanded **\aftergroup** (after the closing of the **\@mkpream** group. All the thick is there: all information collected during the rewritting of X columns (and vertical lines or leaders) can be *reinject*ed into the group below the **\@mkpream** group, by the mean of the **\tabu@mkpreambuffer** (globally defined).
- The private columns types are loaded by **\tabu@rewritefirst**: they will be rewritten afterwards, during the rewritting loop. This way, X column definition for **tabu** are only available during the rewritting process of the **tabu** preamble, making it possible (and easy) to embed a **tabularx** inside a cell of a **tabu**.

- `\save@decl` is modified inside the `\@mkpream` group, if `tabu` is in text mode.

```

988 \tabu@newcolumnntype \tabu@rewritefirst{%
989     \iftabu@long      \aftergroup \tabu@longpream % <the whole implementation is here !>
990     \else              \aftergroup \tabu@pream
991     \fi
992     \let\tabu@        \relax      \let\tabu@hsize      \relax
993     \let\tabu@Xcoefs   \empty     \let\tabu@savels      \relax
994     \tabu@Xcol         \z@        \tabu@cnt            \tw@
995     \gdef\tabu@mkpreambuffer{\tabu@{}}\tabu@measuringfalse
996     \global\setbox\@arstrutbox \box\@arstrutbox
997     \NC@list{\NC@do *}\tabu@textbar \tabu@lines
998     \NC@list\expandafter{\the\NC@list \NC@do X}%
999     \iftabu@siunitx    % <siunitx S and s columns>
1000         \NC@list\expandafter{\the\NC@list \NC@do S\NC@do s}\fi
1001     \NC@list\expandafter{\the\expandafter\NC@list \tabu@highprioritycolumns}%
1002     \expandafter\def\expandafter\tabu@NC@list\expandafter{%
1003         \the\expandafter\NC@list \tabu@NC@list}% % * | X S <original>
1004     \NC@list\expandafter{\expandafter \NC@do \expandafter\usetabu
1005         \expandafter \NC@do \expandafter\preamble
1006         \the\NC@list \NC@do \tabu@rewritemiddle
1007         \NC@do \tabu@rewritelast}%
1008     \tabu@savdecl
1009     \tabu@privatecolumns
1010     \edef\tabu@prev{\the\@temptokena}\NC@find \tabu@rewritemiddle
1011 }% \NC@rewrite@\tabu@rewritefirst

```

`\tabu@rewritemiddle` This new column type is rewritten after `X` columns, because it is declared by when the column  
`\tabu@rewritelast` `\tabu@rewritefirst` is actually rewritten. In the case where `\tabu@target` is  $> 0$  (either because of “`tabu to`” or “`tabu spread`” has been called) and if there is no `X` column, then `@{\extracolsep \@flushglue}` is added at the beginning of the preamble.

To avoid duplicate margin in the `tabu` we have to test the next token in the preamble. If the next token is `|` or `!` then no margin must be added and `@{\extracolsep \@flushglue}` can be inserted at the beginning of the preamble.

Otherwise, we must insert `!{\extracolsep \@flushglue}` in order to keep the margin.

`\tabu@rewritelast` column type is loaded by `\tabu@rewritefirst` column type, only inside the `\@mkpream` group inside the `tabu` environment.

```

1012 \tabu@newcolumnntype \tabu@rewritemiddle{%
1013     \edef\tabu@temp{\the\@temptokena}\NC@find \tabu@rewritelast
1014 }% \NC@rewrite@\tabu@rewritemiddle
1015 \tabu@newcolumnntype \tabu@rewritelast{%
1016     \ifx \tabu@temp\tabu@prev \advance\tabu@cnt \m@ne
1017         \NC@list\expandafter{\tabu@NC@list \NC@do \tabu@rewritemiddle
1018             \NC@do \tabu@rewritelast}%
1019     \else \let\tabu@prev\tabu@temp
1020     \fi
1021     \ifcase \tabu@cnt \expandafter\tabu@endrewrite
1022     \else \expandafter\NC@find \expandafter\tabu@rewritemiddle
1023     \fi
1024 }% \NC@rewrite@\tabu@rewritelast

```



## The end of the rewriting process: determining the tabu strategy

- \tabu@endrewrite** Determines the strategy to be executed **\aftergroup** (at the closing of the **\@mkpream** group):
- 0) There is no real strategy: **tabu** behaves like **tabular**, there no **X** column, and no need to measure the vertical dimensions of the cells (no dynamic spacing, no vertical leader). In case a target has been given to **tabu**, it behaves like **tabular\*** and a infinite stretchability is given to the column inter-space. This is done (if required) by **\tabu@extracolsep**.
  - 1) Measuring natural width of some (or all) columns is compulsory for **tabu spread** of **X** columns with negativ coefficients. Thereafter, the strategy nr 2 will bring into play.
  - 2) Measuring the natural width is not necessary, or has been done before. But **tabu** contains **X** columns and trials have to be performed to reach the desired target, adjusting the **\tabucolX** dimension accordingly. Then, the strategy nr 3 may bring into play, if vertical measure is required.
  - 3) Vertical measure of the cells is required, for vertical spacing adjustment or vertical leaders. This step can be done only if the width are known.
- > 3 The **tabu** is finished and ready to be printed !!

```

1025 \def\tabu@endrewrite {%
1026     \let\tabu@temp \NC@find
1027     \ifx \@arrayright\relax \let\@arrayright \@empty \fi
1028     \count@=%
1029     \ifx \@finalstrut\tabu@finalstrut \z@ % outer in mode 0 print
1030         \iftabu@measuring
1031             \xdef\tabu@mkpreambuffer{\tabu@mkpreambuffer
1032                 \tabu@target          \csname tabu@\the\tabu@nested.T\endcsname
1033                 \tabucolX              \csname tabu@\the\tabu@nested.X\endcsname
1034                 \edef\@halignto {\ifx\@arrayright\@empty to\tabu@target\fi}}%
1035             \fi
1036         \else\iftabu@measuring          4          % X columns
1037             \xdef\tabu@mkpreambuffer{\tabu@{\tabu@mkpreambuffer
1038                 \tabu@target          \the\tabu@target
1039                 \tabu@spreadtarget    \the\tabu@spreadtarget}%
1040                 \def\noexpand\tabu@Xcoefs{\tabu@Xcoefs}%
1041                 \edef\tabu@halignto{\ifx \@arrayright\@empty to\tabu@target\fi}}%
1042             \let\tabu@Xcoefs \relax
1043         \else\ifcase\tabu@nested \thr@@ % outer, no X
1044             \global\let\tabu@afterendpar \relax
1045         \else
1046             \@ne % inner, no X, outer in mode 1 or 2
1047         \fi
1048         \ifdefined\tabu@usetabu
1049         \else \ifdim\tabu@target=\z@
1050         \else \let\tabu@temp \tabu@extracolsep
1051         \fi\fi
1052     \fi
1053     \xdef\tabu@mkpreambuffer{\count@ \the\count@ \tabu@mkpreambuffer}%
1054     \tabu@temp
1055 }% \tabu@endrewrite

```

- \tabu@extracolsep** Inserts **\extracolsep \@flushglue** in front of the preamble, unless another value for **\extracolsep** has been specified.

**\@flushglue** is **0pt plus 1fil**.

```

1056 \def\tabu@extracolsep{\@defaultunits \expandafter\let
1057     \expandafter\tabu@temp \expandafter=\the\@temptokena \relax\@nnil
1058     \ifx \tabu@temp\@sptoken
1059         \expandafter\tabu@gobblespace \expandafter\tabu@extracolsep

```

```

1060 \else
1061 \edef\tabu@temp{\noexpand\NC@find
1062 \if \noexpand\tabu@temp @%
1063 \else\if !\noexpand\tabu@temp @%
1064 \else !%
1065 \fi\fi
1066 {\noexpand\extracolsep\noexpand\@flushglue}}%
1067 \fi
1068 \tabu@temp
1069 }% \tabu@extrac@lsep

```

## 11.10 Implementing the strategy at the exit of the \@mkpream group

### \tabu@select

**\tabu@pream** Triggered `\aftergroup` by the rewriting of `\tabu@rewritefirst`.

The `\tabu@mkpreambuffer` macro is expanded twice: first it injects `\count@` (the strategy number) and `\tabu@nbc`, and redefines itself.

Second – and only if measurements are necessary – it expands into the *trials group* to inject `\tabu@Xcoefs` (the coefficients of *X* columns), `\tabu@Xsum` (the sum of the absolute coefficients), `\tabu@target`, `\tabu@spreadtarget`, and `\tabu@vertical`, which is the number by which one have to increment the strategy number after step 2 (either 1: then a last measure is done for the vertical dimensions, or 255 then the strategy number is > 3 and `\tabu@strategy` orders to finish.)

**\tabu@longpream** This is the *long* version for `longtabu`: the material to collect until `\@preamble` is different !

```

1070 \long\def\tabu@pream #1\@preamble {%
1071 \let\tabu@ \tabu@@ \tabu@mkpreambuffer \tabu@aftergroupcleanup
1072 \NC@list\expandafter {\tabu@NC@list}% in case of nesting...
1073 \ifdefined\tabu@usetabu \tabu@usetabu \tabu@target \z@ \fi
1074 \let\tabu@savedpreamble \@preamble
1075 \global\let\tabu@elapsed\relax
1076 \tabu@thebody ={\#1\tabu@aftergroupcleanup}%
1077 \tabu@thebody =\expandafter{\the\expandafter\tabu@thebody
1078 \@preamble}%
1079 \edef\tabuthethepreamble {\the\tabu@thebody}% ( no @ allowed for \scantokens )
1080 \tabu@select
1081 }% \tabu@pream
1082 \long\def\tabu@longpream #1\LT@bchunk #2\LT@bchunk{%
1083 \let\tabu@ \tabu@@ \tabu@mkpreambuffer \tabu@aftergroupcleanup
1084 \NC@list\expandafter {\tabu@NC@list}% in case of nesting...
1085 \let\tabu@savedpreamble \@preamble
1086 \global\let\tabu@elapsed\relax
1087 \tabu@thebody ={\#1\LT@bchunk #2\tabu@aftergroupcleanup \LT@bchunk}%
1088 \edef\tabuthethepreamble {\the\tabu@thebody}% ( no @ allowed for \scantokens )
1089 \tabu@select
1090 }% \tabu@longpream

```

**\tabu@select** Here we check if trials are required or not: depending on the value of `\count@` (set at `\tabu@endrewrite`, and *injected* here by `\tabu@mkpreambuffer`), on `\iftabu@measuring` (nested trials).

When trials are required, `\tabu@select` give control to `\tabu@setstrategy` (to prepare the neutralisation of commands, save counters etc).

When trials are not required, we just have to expand `\tabuthethepreamble`, after having set up the `\everyrow` stuff properly (for vertical adjustment or vertical measure, if needed).

```

1091 \def\tabu@select {%
1092 \ifnum\tabu@nested>\z@ \tabuscantokensfalse \fi
1093 \ifnum \count@=\@ne \iftabu@measuring \count@=\tw@ \fi\fi
1094 \ifcase \count@

```

```

1095      \global\let\tabu@elapsedtime \relax
1096      \tabu@seteverycr
1097      \expandafter \tabu@hepreamble          % vertical adjustment (inherited from outer)
1098  \or      % exit in vertical measure + struts per cell because no X and outer in mode 3
1099      \tabu@evr{\tabu@verticalinit}\tabu@celllalign@def{\tabu@verticalmeasure}%
1100      \def\tabu@celllalign{\tabu@verticalspacing}%
1101      \tabu@seteverycr
1102      \expandafter \tabu@hepreamble
1103  \or      % exit without measure because no X and outer in mode 4
1104      \tabu@evr{\tabu@celllalign@def{} \let\tabu@celllalign \empty
1105      \tabu@seteverycr
1106      \expandafter \tabu@hepreamble
1107  \else      % needs trials
1108      \tabu@evr{\tabu@celllalign@def{} \let\tabu@celllalign \empty
1109      \tabu@savecounters
1110      \expandafter \tabu@setstrategy
1111  \fi
1112}% \tabu@select
1113 \def\tabu@@ {\gdef\tabu@mkpreambuffer}

```

## General setup for trials: neutralisation of \write etc.

**\tabu@setstrategy** This is the general setup for trials: the tabu will be expanded more than once, thus some protections are set: the value of global counters are saved, footnotes have a special setup, \hbadness and \hfuzz are neutralised etc.

The initial value for \tabucolX is computed with the coefficients stored into \tabu@Wvoeffs:  
\tabu@ {coef1} \tabu@ {coef2} \tabu@ {coef3} etc.

is very suitable for loops on the column width coefficients (without the need of \@for or whatsoever).

```

1114 \def\tabu@setstrategy {\begingroup % <trials group>
1115      \tabu@trialh@@k \tabu@cnt \z@ % number of trials
1116      \hbadness \M \let\hbadness \@tempcnta
1117      \hfuzz \maxdimen \let\hfuzz \@tempdima
1118      \let\write \tabu@nowrite \let\GenericError \tabu@GenericError
1119      \let\savetabu \@gobble \let\tabubefaulttarget \linewidth
1120      \let\@footnotetext \@gobble \let\@xfootnote \tabu@xfootnote
1121      \let\color \tabu@nocolor \let\rowcolor \tabu@norowcolor
1122      \let\tabu@aftergroupcleanup \relax % only after the last trial
1123      \tabu@mkpreambuffer
1124      \ifnum \count@>\thr@@ \let\@halignto \empty \tabucolX@init
1125      \def\tabu@lasttry{\m@ne\p@}\fi
1126      \begingroup \iffalse{\fi \ifnum 0='}\fi
1127      \toks@{}\def\tabu@stack{b}\iftabuscantokens \endlinechar=10 \obeyspaces \fi %
1128      \tabu@collectbody \tabu@strategy %
1129}% \tabu@setstrategy
1130 \def\tabu@savecounters{%
1131      \def\@elt ##1{\csname c@##1\endcsname\the\csname c@##1\endcsname}%
1132      \edef\tabu@clckpt {\begingroup \globaldefs=\@ne \cl@ckpt \endgroup}\let\@elt \relax
1133}% \tabu@savecounters
1134 \def\tabucolX@init {% \tabucolX <= \tabu@target / (sum coefs > 0)
1135      \dimen@ \z@ \tabu@Xsum \z@ \tabucolX \z@ \let\tabu@ \tabu@Xinit \tabu@Xcoefs
1136      \ifdim \dimen@>\z@
1137          \@tempdima \dimexpr \tabu@target *\p@/\dimen@ + \tabu@hfuzz\relax
1138          \ifdim \tabucolX<\@tempdima \tabucolX \@tempdima \fi
1139      \fi
1140}% \tabucolX@init
1141 \def\tabu@Xinit #1#2{\tabu@Xcol #1 \advance \tabu@Xsum
1142      \ifdim #2\p@>\z@ #2\p@ \advance\dimen@ #2\p@

```

```

1143 \else -#2\p@ \tabu@negcoeftrue
1144 \@tempdima \dimexpr \tabu@target*\p@/\dimexpr-#2\p@\relax \relax
1145 \ifdim \tabucolX<\@tempdima \tabucolX \@tempdima \fi
1146 \tabu@wddef{#1}{0pt}%
1147 \fi
1148 }% \tabu@Xinit

```

## Collecting the tabu body

The macro collect the stuff inside `\@array`: depending on the global vertical alignment parameter for the whole tabular, the tabular is built inside a `\vbox`, `\vtop` or `\vcenter` (the default – unless `tabu` is nested).

At this time, we define `\tabu@trial` (which inherits from the `\vbox`, `\vtop` or `\vcenter`) and `\tabu@Xfinish` as well.

**`\tabu@collectbody`** The mechanism is the same as  $\mathcal{AMS}$ -`\collect@body` (also defined in `environ.sty`). The content of the tabular is captured inside `\toks@`, expanded by `\tabu@trial`.

**`\tabu@endofcollect`**

```

1149 \long\def\tabu@collectbody #1#2\end #3{%
1150 \edef\tabu@stack{\tabu@pushbegins #2\begin\end\expandafter\@gobble\tabu@stack}%
1151 \ifx \tabu@stack\@empty
1152 \toks@\expandafter{\expandafter\tabu@thebody\expandafter{\the\toks@ #2}%
1153 \def\tabu@end@envir\end{#3}}%
1154 \iftabuscantokens \def\tabu@endenvir{\let\endarray \@empty
1155 \end{#3}\tabu@gobbleX}%
1156 \else \def\tabu@endenvir\end{#3}}\fi}%
1157 \let\tabu@collectbody \tabu@endofcollect
1158 \else\def\tabu@temp{#3}%
1159 \ifx \tabu@temp\@empty \toks@\expandafter{\the\toks@ #2\end }%
1160 \else \ifx\tabu@temp\tabu@spxiix \toks@\expandafter{\the\toks@ #2\end #3}%
1161 \else \ifx\tabu@temp\tabu@X \toks@\expandafter{\the\toks@ #2\end #3}%
1162 \else \toks@\expandafter{\the\toks@ #2\end{#3}}%
1163 \fi\fi\fi
1164 \fi
1165 \tabu@collectbody{#1}%
1166 }% \tabu@collectbody
1167 \long\def\tabu@pushbegins#1\begin#2\ifx\end#2\else b\expandafter\tabu@pushbegins\fi}%
1168 \def\tabu@endofcollect #1{\ifnum0=‘}\fi
1169 \expandafter\endgroup \the\toks@ #1%
1170 }% \tabu@endofcollect

```

## 11.11 One trial after the other (`\tabu@strategy`)

### Switching between the strategies

**`\tabu@strategy`** This macro does some specific setup depending on the strategy (1, 2 or 3), and orders to finish when all measurements are done.

This consists in a switch (`\ifcase`) which is done before the trials by `\tabu@strategy`, and after the trials by `\tabu@endtrial`.

```

1171 \def\tabu@strategy {\relax % stops \count@ assignment !
1172 \ifcase\count@ % case 0 = print with vertical adjustment (outer is finished)
1173 \expandafter \tabu@endoftrials
1174 \or % case 1 = exit in vertical measure (outer in mode 3)
1175 \expandafter\xdef\csname tabu@\the\tabu@nested.T\endcsname{\the\tabu@target}%
1176 \expandafter\xdef\csname tabu@\the\tabu@nested.X\endcsname{\the\tabucolX}%
1177 \expandafter \tabu@endoftrials
1178 \or % case 2 = exit with a rule replacing the table (outer in mode 4)
1179 \expandafter \tabu@quickend
1180 \or % case 3 = outer is in mode 3 because of no X
1181 \begingroup

```

```

1182         \tabu@evr{\tabu@verticalinit}\tabu@celllalign@def{\tabu@verticalmeasure}%
1183         \def\tabu@cellralign{\tabu@verticalspacing}%
1184         \expandafter \tabu@measuring
1185     \else                                     % case 4 = horizontal measure
1186         \begingroup
1187             \global\let\tabu@elapsedtime \tabu@message@etime
1188             \long\def\multicolumn##1##2##3{\multispan{##1}}%
1189             \let\tabu@startpboxORI \startpbox
1190             \iftabu@spread
1191                 \def\tabu@naturalXmax {\z@}%
1192                 \let\tabu@naturalXmin \tabu@naturalXmax
1193                 \tabu@evr{\global\tabu@naturalX \z@}%
1194                 \let\startpbox \tabu@startpboxmeasure
1195             \else\iftabu@negcoef
1196                 \let\startpbox \tabu@startpboxmeasure
1197             \else \let\startpbox \tabu@startpboxquick
1198             \fi\fi
1199             \expandafter \tabu@measuring
1200     \fi
1201 }% \tabu@strategy

```

**\tabu@measuring** Expands \tabu@trial with the whole content of the environment stored in \toks@ by \tabu@collectbody. At the end of the trial, \count@ will be reassigned to the value it had before the trial. Then \tabu@endtrial will choose the algorithm depending on the strategy number, and set the new strategy number (into \count@ again) for the next step.

**\tabu@trial** This is the starting point of trials: \halign is expanded here.

**\tabu@longtrial** This is the long version of \tabu@trial for longtabu. Almost the same apart for the math group and the end (a longtable environment does not finish with \endarray).

```

1202 \def\tabu@measuring{\expandafter \tabu@trial \expandafter
1203                     \count@ \the\count@ \tabu@endtrial
1204 }% \tabu@measuring
1205 \def\tabu@trial{\iftabu@long \tabu@longtrial \else \tabu@shorttrial \fi}
1206 \def\tabu@shorttrial {\setbox\tabu@box \hbox\bgroup \tabu@seteverycr
1207     \ifx \tabu@savecounters\relax \else
1208         \let\tabu@savecounters \relax \tabu@clckpt \fi
1209     $\iftabuscantokens \tabu@rescan \else \expandafter\@secondoftwo \fi
1210     \expandafter{\expandafter \tabu@thepreamble
1211                 \the\tabu@thebody
1212                 \cename tabu@adl@endtrial\endcename
1213                 \endarray}$\egroup % got \tabu@box
1214 }% \tabu@shorttrial
1215 \def\tabu@longtrial {\setbox\tabu@box \hbox\bgroup \tabu@seteverycr
1216     \ifx \tabu@savecounters\relax \else
1217         \let\tabu@savecounters \relax \tabu@clckpt \fi
1218     \iftabuscantokens \tabu@rescan \else \expandafter\@secondoftwo \fi
1219     \expandafter{\expandafter \tabu@thepreamble
1220                 \the\tabu@thebody
1221                 \LT@echunk
1222                 \global\setbox\@ne \hbox{\unhbox\@ne}\kern\wd\@ne
1223                 \LT@get@widths}\egroup % got \tabu@box
1224 }% \tabu@longtrial
1225 \def\tabu@adl@endtrial{% <arydshln in nested trials - problem for global column counters!>
1226     \crcr \noalign{\global\adl@ncol \tabu@nbc}}% anything global is crap, junky and fails !

```

**\tabu@seteverycr** \ialign resets \everycr to an empty token. This macro sets \everycr for the tabu environment : a bridge around \ialign is built: \everycr redefines itself \afterassignment!

```

1227 \def\tabu@seteverycr {\tabu@reset
1228     \everycr \expandafter{\the\everycr \tabu@everycr}%
1229     \let\everycr \tabu@noeverycr % <for ialign>
1230 }% \tabu@seteverycr
1231 \def\tabu@noeverycr{\aftergroup\tabu@restoreeverycr \afterassignment}\toks@}
1232 \def\tabu@restoreeverycr {\let\everycr \tabu@@everycr}
1233 \def\tabu@everycr {\iftabu@everyrow \noalign{\tabu@everyrow}\fi}

```

**\tabu@endoftrials** When the algorithm said the tabular was ready to be printed, **\tabu@endoftrials** closes the trials group and prints the tabular...

The required values (column widths, struts etc.) are *injected* into the group by the mean of the buffer **\tabu@bufferX** (locally defined).

**\tabu@closetrialsgroup** This closes the group in which all the trials are done.

```

1234 \def\tabu@endoftrials {%
1235     \iftabuscantokens \expandafter\@firstoftwo
1236     \else \expandafter\@secondoftwo
1237     \fi
1238     {\expandafter \tabu@closetrialsgroup \expandafter
1239     \tabu@rescan \expandafter{%
1240         \expandafter\tabuthepreamble
1241         \the\tabu@thebody
1242         \endarray}}
1243     {\expandafter\tabu@closetrialsgroup \expandafter
1244     \tabuthepreamble
1245     \the\tabu@thebody}%
1246     \tabu@endenvir % Finish !
1247 }% \tabu@endoftrials
1248 \def\tabu@closetrialsgroup {%
1249     \toks@\expandafter{\tabu@endenvir}%
1250     \edef\tabu@bufferX{\endgroup
1251         \tabucolX \the\tabucolX
1252         \tabu@target \the\tabu@target
1253         \tabu@cnt \the\tabu@cnt
1254         \def\noexpand\tabu@endenvir{\the\toks@}%
1255         %Quid de \@halignto = \tabu@halignto ??
1256     }% \tabu@bufferX
1257     \tabu@bufferX
1258     \ifcase\tabu@nested % print out (outer in mode 0)
1259         \global\tabu@cnt \tabu@cnt
1260         \tabu@evr{\tabu@verticaldynamicadjustment}%
1261         \tabu@celllalign@def{\everypar{}}\let\tabu@cellralign \@empty
1262         \let\@finalstrut \tabu@finalstrut
1263     \else % vertical measure of nested tabu
1264         \tabu@evr{\tabu@verticalinit}%
1265         \tabu@celllalign@def{\tabu@verticalmeasure}%
1266         \def\tabu@cellralign{\tabu@verticalspacing}%
1267     \fi
1268     \tabu@clckpt \let\@halignto \tabu@halignto
1269     \let\@halignto \@empty
1270     \tabu@seteverycr
1271     \ifdim \tabustrutrule>\z@ \ifnum\tabu@nested=\z@
1272         \setbox\@arstrutbox \box\voidb@x % force \@arstrutbox to be rebuilt (visible struts)
1273     \fi\fi
1274 }% \tabu@closetrialsgroup

```

**\tabu@quickend** Quick exit after having measuring the natural width of a nested **tabu**.

```

1275 \def\tabu@quickend {\expandafter \endgroup \expandafter

```

```

1276 \tabu@target \the\tabu@target \tabu@quickrule
1277 \let\endarray \relax \tabu@endenvir
1278 }% \tabu@quickend

\tabu@endtrial Depending on the strategy that was just applied, \tabu@endtrial chooses the algorithm and determines
the number of the strategy for the next step.

1279 \def\tabu@endtrial {\relax % stops \count@ assignment !
1280 \ifcase \count@ \tabu@err % case 0 = impossible here
1281 \or \tabu@err % case 1 = impossible here
1282 \or \tabu@err % case 2 = impossible here
1283 \or % case 3 = outer goes into mode 0
1284 \def\tabu@bufferX{\endgroup}\count@ \z@
1285 \else % case 4 = outer goes into mode 3
1286 \iftabu@spread \tabu@spreadarith % inner into mode 1 (outer in mode 3)
1287 \else \tabu@arith % or 2 (outer in mode 4)
1288 \fi
1289 \count@=%
1290 \ifcase\tabu@nested \thr@@ % outer goes into mode 3
1291 \else\iftabu@measuring \tw@ % outer is in mode 4
1292 \else \@ne % outer is in mode 3
1293 \fi\fi
1294 \edef\tabu@bufferX{\endgroup
1295 \tabucolX \the\tabucolX
1296 \tabu@target \the\tabu@target}%
1297 \fi
1298 \expandafter \tabu@bufferX \expandafter
1299 \count@ \the\count@ \tabu@strategy
1300 }% \tabu@endtrial
1301 \def\tabu@err{\errmessage{(tabu) Internal impossible error! (\count@=\the\count@)}}

```

## 11.12 The algorithms: Measuring the tabu box

At the end of each trial, we call `\tabu@arith` (or `\tabu@spreadarith`) to computes the widths and update the values.

At the exit, `\iftabu@measuring` is set to `\iftrue`: a further trial is necessary, or `\iffalse`: the target width is reached.

### The arithmetic of X columns: the tabu to case

`\tabu@arithnegcoef` This is a loop against the width coefficients. There is no `\@for` or `\@whiles` because `\tabu@Xcoefs` stores the series in the form: `\tabu@ {coef1} \tabu@ {coef2} \tabu@ {coef3}`.

Thus, just `\let \tabu@` to be `\tabu@arith@negcoef` and expand `\tabu@Xcoefs`!

The aim of the game is to *neutralize* some X columns: when their natural width are less than `coef×\tabucolX`.

```

1302 \def\tabu@arithnegcoef {%
1303 \tempdima \z@ \dimen@ \z@ \let\tabu@ \tabu@arith@negcoef \tabu@Xcoefs
1304 }% \tabu@arithnegcoef
1305 \def\tabu@arith@negcoef #1#2{%
1306 \ifdim #2\p@>\z@ \advance\dimen@ #2\p@ % saturated by definition
1307 \advance\tempdima #2\tabucolX
1308 \else
1309 \ifdim -#2\tabucolX <\tabu@wd{#1}% c_i X < natural width <= \tabu@target-> saturated
1310 \advance\dimen@ -#2\p@
1311 \advance\tempdima -#2\tabucolX
1312 \else
1313 \advance\tempdima \tabu@wd{#1}% natural width <= c_i X => neutralised
1314 \ifdim \tabu@wd{#1}<\tabu@target \else % neutralised
1315 \advance\dimen@ -#2\p@ % saturated (natural width = tabu@target)

```



```

1316
1317         \fi
1318     \fi
1319 }% \tabu@arith@negcoef

```

`\tabu@arith` General algorithms for `\tabu` to with X columns.

```

1320 \def\tabu@givespace #1#2{% here \tabu@DELTA < \z@
1321     \ifdim \@tempdima=\z@
1322         \tabu@wddef{#1}{\the\dimexpr -\tabu@DELTA*\p@\tabu@Xsum}%
1323     \else
1324         \tabu@wddef{#1}{\the\dimexpr \tabu@hsizex{#1}{#2}
1325             *(\p@-\tabu@DELTA*\p@/\@tempdima)/\p@\relax}%
1326     \fi
1327 }% \tabu@givespace
1328 \def\tabu@arith {\advance\tabu@cmt \@ne
1329     \ifnum \tabu@cmt=\@ne \tabu@message{\tabu@titles}\fi
1330     \tabu@arithnegcoef
1331     \@tempdimb \dimexpr \wd\tabu@box -\@tempdima \relax % <incompressible material>
1332     \tabu@DELTA = \dimexpr \wd\tabu@box - \tabu@target \relax
1333     \tabu@message{\tabu@message@arith}%
1334     \ifdim \tabu@DELTA <\tabu@hfuzz
1335         \ifdim \tabu@DELTA<\z@           % wd (\tabu) <\tabu@target ?
1336             \let\tabu@ \tabu@givespace \tabu@Xcoeffs
1337             \advance\@tempdima \@tempdimb \advance\@tempdima -\tabu@DELTA % for message
1338             \else    % already converged: nothing to do but nearly impossible...
1339                 \fi
1340             \tabucolX \maxdimen
1341             \tabu@measuringfalse
1342         \else                                     % need for narrower X columns
1343             \tabucolX =\dimexpr (\@tempdima -\tabu@DELTA) *\p@\tabu@Xsum \relax
1344             \tabu@measuringtrue
1345             \@whilesw \iftabu@measuring\fi {%
1346                 \advance\tabu@cmt \@ne
1347                 \tabu@arithnegcoef
1348                 \tabu@DELTA =\dimexpr \@tempdima+\@tempdimb -\tabu@target \relax % always < 0 here
1349                 \tabu@message{\tabu@header
1350                     \tabu@msgalign \tabucolX { }{ }{ }{ }{ }{ }{ }{ }\@@
1351                     \tabu@msgalign \@tempdima+\@tempdimb { }{ }{ }{ }{ }{ }{ }{ }\@@
1352                     \tabu@msgalign \tabu@target { }{ }{ }{ }{ }{ }{ }{ }\@@
1353                     \tabu@msgalign@PT \dimen@ { }{ }{ }{ }{ }{ }{ }{ }\@@
1354                     \ifdim -\tabu@DELTA<\tabu@hfuzz \tabu@spaces target ok\else
1355                         \tabu@msgalign \dimexpr -\tabu@DELTA *\p@/\dimen@ { }{ }{ }{ }{ }{ }{ }{ }\@@
1356                     \fi}%
1357                 \ifdim -\tabu@DELTA<\tabu@hfuzz
1358                     \advance\@tempdima \@tempdimb % for message
1359                     \tabu@measuringfalse
1360                 \else
1361                     \advance\tabucolX \dimexpr -\tabu@DELTA *\p@/\dimen@ \relax
1362                 \fi
1363             }%
1364     \fi
1365     \tabu@message{\tabu@message@reached}%
1366     \edef\tabu@bufferX{\endgroup \tabu@cmt       \the\tabu@cmt
1367                          \tabucolX              \the\tabucolX
1368                          \tabu@target            \the\tabu@target}%
1369 }% \tabu@arith

```

## The arithmetic of X columns for tabu spread

`\tabu@spreadarith` Algorithm for `\tabu spread` with `X` columns: the aim of the game is to compute the target (relative to the natural width of the tabular) and go to `\tabu@arith` afterwards.

```

1370 \def\tabu@spreadarith {%
1371     \dimen@ \z@ \@tempdima \tabu@naturalXmax \let\tabu@ \tabu@spread@arith \tabu@Xcoefs
1372     \edef\tabu@naturalXmin {\the\dimexpr\tabu@naturalXmin*\dimen@/\p@}%
1373     \@tempdimc =\dimexpr \wd\tabu@box -\tabu@naturalXmax+\tabu@naturalXmin \relax
1374     \iftabu@measuring
1375         \tabu@target =\dimexpr \@tempdimc+\tabu@spreadtarget \relax
1376         \edef\tabu@bufferX{\endgroup \tabucolX \the\tabucolX \tabu@target\the\tabu@target}%
1377     \else
1378         \tabu@message{\tabu@message@spreadarith}%
1379         \ifdim \dimexpr \@tempdimc+\tabu@spreadtarget >\tabu@target
1380             \tabu@message{(tabu) spread
1381                 \ifdim \@tempdimc>\tabu@target useless here: default target used%
1382                 \else too large: reduced to fit default target\fi.}%
1383         \else
1384             \tabu@target =\dimexpr \@tempdimc+\tabu@spreadtarget \relax
1385             \tabu@message{(tabu) spread: New target set to \the\tabu@target^^J}%
1386         \fi
1387         \begingroup \let\tabu@wddef \@gobbletwo
1388             \@tempdimb \@tempdima
1389             \tabucolX@init
1390             \tabu@arithnegcoef
1391             \wd\tabu@box =\dimexpr \wd\tabu@box +\@tempdima-\@tempdimb \relax
1392             \expandafter\endgroup \expandafter\tabucolX \the\tabucolX
1393             \tabu@arith
1394         \fi
1395     }% \tabu@spreadarith
1396 \def\tabu@spread@arith #1#2{%
1397     \ifdim #2\p@>\z@ \advance\dimen@ #2\p@
1398     \else
1399         \advance\@tempdima \tabu@wd{#1}\relax
1400     }% \tabu@spread@arith

```

### Reporting in the .log file (debugshow option)

```
\tabu@message@defaulttarget
```

```

1401 \def\tabu@message@defaulttarget{%
1402     \ifnum\tabu@nested=\z@^^J(tabu) Default target:
1403     \ifx\tabudefaulttarget\linewidth \string\linewidth
1404     \ifdim \tabu@thetarget=\linewidth \else
1405         -\the\dimexpr\linewidth-\tabu@thetarget\fi =
1406     \else\ifx\tabudefaulttarget\linegoal\string\linegoal=
1407     \fi\fi
1408     \else (tabu) Default target (nested): \fi
1409     \the\tabu@target \on@line
1410     \ifnum\tabu@nested=\z@ , page \the\c@page\fi}
1411 \def\tabu@message@target {^^J(tabu) Target specified:
1412     \the\tabu@target \on@line, page \the\c@page}

```

`\tabu@message@arith`

```

1413 \def\tabu@message@arith {\tabu@header
1414     \tabu@msgalign \tabucolX { }{ }{ }{ }{ }{\@@}
1415     \tabu@msgalign \wd\tabu@box { }{ }{ }{ }{ }{\@@}
1416     \tabu@msgalign \tabu@target { }{ }{ }{ }{ }{\@@}
1417     \tabu@msgalign@PT \dimen@ { }{}{}{}{}{}{}{}{}{}{}{}{}{\@@}

```

```

1418      \ifdim \tabu@DELTA<\tabu@hfuzz giving space\else
1419      \tabu@msgalign \dimexpr (\@tempdima-\tabu@DELTA) *\p@/\tabu@Xsum -\tabu@colX {}{}{}{}{}{}\\@@
1420      \fi
1421 }% \tabu@message@arith

```

```
\tabu@message@spreadarith
```

```

1422 \def\tabu@message@spreadarith {\tabu@spreadheader
1423   \tabu@msgalign \tabu@spreadtarget { }{ }{ }{ }}\@@
1424   \tabu@msgalign \wd\tabu@box { }{ }{ }{ }}\@@
1425   \tabu@msgalign -\tabu@naturalXmax { }{ }{ }{ }}\@@
1426   \tabu@msgalign \tabu@naturalXmin { }{ }{ }{ }}\@@
1427   \tabu@msgalign \ifdim \dimexpr \@tempdimc>\tabu@target \tabu@target
1428                       \else \@tempdimc+\tabu@spreadtarget \fi
1429   {}{}{}{}{}\@@}

```

\tabu@message@negcoef

```

1430 \def\tabu@message@negcoef #1#2{
1431     \tabu@spaces\tabu@spaces\space * #1. X[\rem@pt#2]:
1432     \space width = \tabu@wd {#1}
1433     \expandafter\string\csname tabu@\the\tabu@nested.W\number#1\endcsname
1434     \ifdim -\tabu@pt#2\tabucolX<\tabu@target
1435     < \number-\rem@pt#2 X
1436     = \the\dimexpr -\tabu@pt#2\tabucolX \relax
1437     \else
1438     <= \the\tabu@target\space < \number-\rem@pt#2 X\fi}

```

\tabu@message@reached

```

1439 \def\tabu@message@reached{\tabu@header
1440      ***** Reached Target:
1441      hfuzz = \tabu@hfuzz\on@line\space *****}

```

\tabu@message@etime

```

1442 \def\tabu@message@etime{\edef\tabu@stoptime{\the\pdfelapsedetime}%
1443 \tabu@message{(tabu)\tabu@spaces Time elapsed during measure:
1444 \the\numexpr(\tabu@stoptime-\tabu@starttime-32767)/65536\relax sec
1445 \the\numexpr\numexpr(\tabu@stoptime-\tabu@starttime)
1446 -\numexpr(\tabu@stoptime-\tabu@starttime-32767)/65536\relax*65536\relax
1447 *1000/65536\relax ms \tabu@spaces(\the\tabu@cnt\space
1448 cycle\ifnum\tabu@cnt>\@ne s\fi)^J^J}}

```

\tabu@message@verticalsp

```

1449 \def\tabu@message@verticalsp {%
1450     \ifdim \@tempdima>\tabu@ht
1451         \ifdim \@tempdimb>\tabu@dp
1452             \expandafter\expandafter\expandafter\string\tabu@ht =
1453                 \tabu@msgalign \@tempdima { }{ }{ }{ }{ }{ }\@@
1454             \expandafter\expandafter\expandafter\string\tabu@dp =
1455                 \tabu@msgalign \@tempdimb { }{ }{ }{ }{ }{ }\@@^^J%
1456         \else
1457             \expandafter\expandafter\expandafter\string\tabu@ht =
1458                 \tabu@msgalign \@tempdima { }{ }{ }{ }{ }{ }\@@^^J%
1459         \fi
1460     \else\ifdim \@tempdimb>\tabu@dp
1461         \tabu@spaces\tabu@spaces\tabu@spaces
1462         \expandafter\expandafter\expandafter\string\tabu@dp =
1463             \tabu@msgalign \@tempdimb { }{ }{ }{ }{ }{ }\@@^^J\fi
1464     \fi
1465 }% \tabu@message@verticalsp

```

\tabu@message@save

```

1466 \edef\tabu@spaces{\@spaces}
1467 \def\tabu@strippt{\expandafter\tabu@pt\the}
1468 {\@makeother\P \@makeother{T\lowercase{\gdef\tabu@pt #1PT{#1}}}%
1469 \def\tabu@msgalign{\expandafter\tabu@msg@align\the\dimexpr}
1470 \def\tabu@msgalign@PT{\expandafter\tabu@msg@align\romannumeral-\`0\tabu@strippt}
1471 \def\do #1{%
1472     \def\tabu@msg@align##1.##2##3##4##5##6##7##8##9\@@{%
1473         \ifnum##1<10 #1 #1\else
1474             \ifnum##1<100 #1 \else
1475                 \ifnum##1<\@m #1\fi\fi\fi
1476                 ##1.##2##3##4##5##6##7##8##1}%
1477     \def\tabu@header{(tabu) \ifnum\tabu@cnt<10 #1\fi\the\tabu@cnt) }%
1478     \def\tabu@titles{\ifnum \tabu@nested=\z@
1479         (tabu) Try#1 #1 tabu X #1 #1 #1tabu Width #1 #1 Target
1480             #1 #1 #1 Coefs #1 #1 #1 Update^^J\fi}%
1481     \def\tabu@spreadheader{%
1482         (tabu) Try#1 #1 Spread #1 #1 tabu Width #1 #1 #1 Nat. X #1 #1 #1 #1Nat. Min.
1483             #1 New Target^^J%
1484         (tabu) sprd}
1485     \def\tabu@message@save {\begingroup
1486         \def\x #####1{\tabu@msg@align #####1{ }}{ }{ }{ }{ }\@@}
1487         \def\z #####1{\expandafter\x\expandafter{\romannumeral-\`0\tabu@strippt
1488             \dimexpr#####1\p@{ }}{ }{ }{ }{ }%
1489         \let\color \relax \def\tabu@rulesstyle #####1####2{\detokenize{#####1}}%
1490         \let\CT@arc@ \relax \let\@preamble \@gobble
1491         \let\tabu@savedit@firstofone
1492         \let\tabu@savedit@params \@firstofone
1493         \def\tabu@target #####1\relax {(tabu) target #1 #1 #1 #1 #1 = \x{#####1}^^J}%
1494         \def\tabu@colX #####1\relax {(tabu) X columns width#1 = \x{#####1}^^J}%
1495         \def\tabu@nbcols #####1\relax {(tabu) Number of columns: \z{#####1}^^J}%
1496         \def\tabu@aligndefault #####1{(tabu) Default alignment: #1 #1 #####1^^J}%
1497         \def\colsep #####1\relax {(tabu) column sep #1 #1 #1 = \x{#####1}^^J}%
1498         \def\arrayrulewidth #####1\relax{(tabu) arrayrulewidth #1 = \x{#####1}}%
1499         \def\doublerulesep #####1\relax { doublerulesep = \x{#####1}^^J}%
1500         \def\extratabsurround#####1\relax{(tabu) extratabsurround = \x{#####1}^^J}%
1501         \def\extrarowheight #####1\relax{(tabu) extrarowheight #1 = \x{#####1}}%
1502         \def\extrarowdepth #####1\relax {extrarowdepth = \x{#####1}^^J}%
1503         \def\arraystretch #####1{(tabu) arraystretch #1 #1 = \z{#####1}^^J}%
1504         \def\minrowclearance#####1\relax{(tabu) minrowclearance #1 = \x{#####1}^^J}%
1505         \def\tabu@arc@L #####1{(tabu) taburulecolor #1 #1 = #####1^^J}%
1506         \def\tabu@drsc@L #####1{(tabu) tabudoublerulecolor= #####1^^J}%
1507         \def\tabu@evr@L #####1{(tabu) everyrow #1 #1 #1 #1 = \detokenize{#####1}^^J}%
1508         \def\tabu@ls@L #####1{(tabu) line style = \detokenize{#####1}^^J}%
1509         \def\NC@find #####1\@nil{(tabu) tabu preamble#1 #1 = \detokenize{#####1}^^J}%
1510         \def\tabu@wddedf#####1####2{(tabu) Natural width #####1 = \x{#####2}^^J}%
1511         \let\edef \@gobbletwo \let\def \@empty \let\let \@gobbletwo
1512         \tabu@message{%
1513             (tabu) \string\savetabu{\tabu@temp}: \oneline^^J%
1514             \tabu@usetabu \@nil^^J}%
1515         \endgroup}
1516 }\do{ }

```

## 11.13 Measuring the natural width of columns (**varwidth** code from D. Arseneau)

**\tabu@startpboxmeasure** The important job is done at the end: by **\tabu@endpboxmeasure**.

When “**tabu spread**” is used with **X** columns, the first trial must measure the natural width of the columns. When **X** columns have negativ coefficient, the natural is computed after the target has been reached, with the absolute coefficients.

Nested trials may occur (**tabu spread** inside a **X** column with negativ coefficient for example).

For the furthur trials, the standard scheme for **X** column is used: the natural width is measured only once.

pdfT<sub>E</sub>X font expansion is disabled inside the **varwidth** environment (we set **\pdfadjustspacing** to 0).

```

1517 \def\tabu@startpboxmeasure #1{\bgroup % entering \vtop
1518     \edef\tabu@temp{\expandafter\@secondoftwo \ifx\tabu@hsize #1\else\relax\fi}%
1519     \ifodd 1\ifx \tabu@temp\@empty 0 \else % starts with \tabu@hsize ?
1520         \iftabu@spread \else % if spread -> measure
1521             \ifdim \tabu@temp\p@>\z@ 0 \fi\fi\fi% if coef>0 -> do not measure
1522             \let\@startpbox \tabu@startpboxORI % restore immediately (nesting)
1523             \tabu@measuringtrue % for the quick option...
1524             \tabu@Xcol =\expandafter\@firstoftwo\ifx\tabu@hsize #1\fi
1525             \ifdim \tabu@temp\p@>\z@ \ifdim \tabu@temp\tabucolX<\tabu@target
1526                 \tabu@target=\tabu@temp\tabucolX \fi\fi
1527             \setbox\tabu@box \hbox \bgroup
1528                 \begin{varwidth}\tabu@target
1529                     \let\FV@ListProcessLine \tabu@FV@ListProcessLine % \hbox to natural width...
1530                     \narrowragged \arraybackslash \parfillskip \@flushglue
1531                     \ifdefined\pdfadjustspacing \pdfadjustspacing\z@ \fi
1532                     \bgroup \aftergroup\tabu@endpboxmeasure
1533                     \ifdefined \cellspacetoplimit \tabu@cellspacepatch \fi
1534             \else \expandafter\@gobble
1535                 \tabu@startpboxquick{#1}% \@gobble \bgroup
1536             \fi
1537 }% \tabu@startpboxmeasure
1538 \def\tabu@cellspacepatch{\def\bcolumn##1\@nil{}\let\ecolumn\@empty
1539     \bgroup\color@begingroup}

```

**\tabu@endpboxmeasure** The cell has been built inside a box: we have to get its dimensions, and update **\tabu@naturalX**, **\tabu@naturalXmin** and **\tabu@naturalXmax** accordingly (for **tabu spread**), and even store (globally) each column width: the column width is the maximum width of the cells it contains.

```

1540 \def\tabu@endpboxmeasure {%
1541     \@finalstrut \@arstrutbox
1542     \end{varwidth}\egroup % <got my \tabu@box>
1543     \ifdim \tabu@temp\p@<\z@ % neg coef
1544         \ifdim \tabu@wd\tabu@Xcol <\wd\tabu@box
1545             \tabu@wddef\tabu@Xcol {\the\wd\tabu@box}%
1546             \tabu@debug{\tabu@message@endpboxmeasure}%
1547         \fi
1548     \else % spread coef>0
1549         \global\advance \tabu@naturalX \wd\tabu@box
1550         \@tempdima =\dimexpr \wd\tabu@box * \p@/\dimexpr \tabu@temp\p@\relax \relax
1551         \ifdim \tabu@naturalXmax <\tabu@naturalX
1552             \xdef\tabu@naturalXmax {\the\tabu@naturalX}\fi
1553         \ifdim \tabu@naturalXmin <\@tempdima
1554             \xdef\tabu@naturalXmin {\the\@tempdima}\fi
1555     \fi
1556     \box\tabu@box \egroup % end of \vtop (measure) restore \tabu@target
1557 }% \tabu@endpboxmeasure
1558 \def\tabu@wddef #1{\expandafter\xdef

```

```

1559 \csname tabu@the\tabu@nested.W\number#1\endcsname}
1560 \def\tabu@wd #1{\csname tabu@the\tabu@nested.W\number#1\endcsname}
1561 \def\tabu@message@endpboxmeasure{\tabu@spaces\tabu@spaces<-> % <-> save natural wd
1562 \the\tabu@Xcol. X[\tabu@temp]:
1563 target = \the\tabucolX \space
1564 \expandafter\expandafter\expandafter\string\tabu@wd\tabu@Xcol
1565 =\tabu@wd\tabu@Xcol
1566 }% \tabu@message@endpboxmeasure

```

**\tabu@startpboxquick** Contents of paragraph columns are not built during trials in strategy number 4.

```

1567 \def\tabu@startpboxquick {\bgroup
1568 \let\@startpbox \tabu@startpboxORI % restore immediately
1569 \let\tabu \tabu@quick % \begin is expanded before...
1570 \expandafter\@gobble \@startpbox % gobbles \bgroup
1571 }% \tabu@startpboxquick
1572 \def\tabu@quick {\begingroup \iffalse{\fi \ifnum0='}\fi
1573 \toks@{\def\tabu@stack{b}\tabu@collectbody \tabu@endquick
1574 }% \tabu@quick
1575 \def\tabu@endquick {%
1576 \ifodd 1\ifx\tabu@end@envir\tabu@endtabu \else
1577 \ifx\tabu@end@envir\tabu@endtabus \else 0\fi\fi\relax
1578 \endgroup
1579 \else \let\endtabu \relax
1580 \tabu@end@envir
1581 \fi
1582 }% \tabu@quick
1583 \def\tabu@endtabu {\end{tabu}}
1584 \def\tabu@endtabus {\end{tabu*}}

```

## 11.14 Measuring the height and depths of rows

**\tabu@verticalmeasure** Starting point for vertical measure of every cell. Only the maxima/minima are stored, for  $\mathcal{T}_N b c$  must know the height/depth of every row.

```

1585 \def\tabu@verticalmeasure{\everypar{}}%
1586 \ifnum \currentgrouptype>12 % 14=semi-simple, 15=math shift group
1587 \setbox\tabu@box =\hbox\bgroup
1588 \let\tabu@verticalspacing \tabu@verticalsp@lcr
1589 \d@llarbegin % after \hbox ...
1590 \else
1591 \edef\tabu@temp{\ifnum\currentgrouptype=5\vtop
1592 \else\ifnum\currentgrouptype=12\vcenter
1593 \else\vbox\fi\fi}%
1594 \setbox\tabu@box \hbox\bgroup$\tabu@temp \bgroup
1595 \let\tabu@verticalspacing \tabu@verticalsp@pmb
1596 \fi
1597 }% \tabu@verticalmeasure

```

**\tabu@verticalsp@lcr** Vertical spacing adjustment for standard l, c, r columns.

```

1598 \def\tabu@verticalsp@lcr{%
1599 \d@llarend \egroup % <got my \tabu@box>
1600 \@tempdima \dimexpr \ht\tabu@box+\abovetabulinesep
1601 \@tempdimb \dimexpr \dp\tabu@box+\belowtabulinesep \relax
1602 \ifdim\tabustrutrule>\z@ \tabu@debug{\tabu@message@verticalsp}\fi
1603 \ifdim \tabu@ht<\@tempdima \tabu@htdef{\the\@tempdima}\fi
1604 \ifdim \tabu@dp<\@tempdimb \tabu@dpdef{\the\@tempdimb}\fi
1605 \noindent\vrule height\@tempdima depth\@tempdimb
1606 }% \tabu@verticalsp@lcr

```

**\tabu@verticalsp@pmb** Vertical spacing adjustment with struts for p, m, or b columns.

```

1607 \def\tabu@verticalsp@pmb{% inserts struts as needed
1608     \par \expandafter\egroup
1609         \expandafter$\expandafter
1610             \egroup \expandafter
1611                 \@tempdimc \the\prevdepth
1612     \@tempdima \dimexpr \ht\tabu@box+\abovetabulinesep
1613     \@tempdimb \dimexpr \dp\tabu@box+\belowtabulinesep \relax
1614     \ifdim\tabustrutrule>\z@ \tabu@debug{\tabu@message@verticalsp}\fi
1615     \ifdim \tabu@ht<\@tempdima \tabu@htdef{\the\@tempdima}\fi
1616     \ifdim \tabu@dp<\@tempdimb \tabu@dpdef{\the\@tempdimb}\fi
1617     \let\@finalstrut \@gobble
1618     \hrule height\@tempdima depth\@tempdimb width\hsize
1619 %% \box\tabu@box
1620 }% \tabu@verticalsp@pmb

```

**\tabu@verticalinit** Initialisation of \tabu@ht and \tabu@dp. Done at **\everyrow**.

```

1621 \def\tabu@verticalinit{%
1622     \ifnum \c@taburow=\z@ \tabu@rearstrut \fi % after \tabu@reset !
1623     \advance\c@taburow \@ne
1624     \tabu@htdef{\the\ht\@arstrutbox}\tabu@dpdef{\the\dp\@arstrutbox}%
1625     \advance\c@taburow \m@ne
1626 }% \tabu@verticalinit
1627 \def\tabu@htdef {\expandafter\xdef \csname tabu@the\tabu@nested.H\the\c@taburow\endcsname}
1628 \def\tabu@ht {\csname tabu@the\tabu@nested.H\the\c@taburow\endcsname}
1629 \def\tabu@dpdef {\expandafter\xdef \csname tabu@the\tabu@nested.D\the\c@taburow\endcsname}
1630 \def\tabu@dp {\csname tabu@the\tabu@nested.D\the\c@taburow\endcsname}

```

**\tabu@verticaldynamicadjustment** This updates the \@arstrutbox at **\everyrow** (*ie.* **\everycr**) in order to adjust the vertical spacing of cells.

```

1631 \def\tabu@verticaldynamicadjustment {%
1632     \advance\c@taburow \@ne
1633     \extrarowheight \dimexpr\tabu@ht - \ht\strutbox
1634     \extrarowdepth \dimexpr\tabu@dp - \dp\strutbox
1635     \let\arraystretch \@empty
1636     \advance\c@taburow \m@ne
1637 }% \tabu@verticaldynamicadjustment

```

## 11.15 \tabuphantomline

**\tabuphantomline** This macro inserts a phantom line in front of a tabu. This is necessary when you use **\usetabu** with tabu X column, with a single line containing **\multicolumn...**

```

1638 \def\tabuphantomline{\cr cr \noalign{%
1639     {\globaldefs \@ne
1640         \setbox\@arstrutbox \box\voidb@x
1641         \let\tabu@celllalign \tabu@celllalign
1642         \let\tabu@cellralign \tabu@cellralign
1643         \let\tabu@celllleft \tabu@celllleft
1644         \let\tabu@cellrright \tabu@cellrright
1645         \let\tabu@thevline \tabu@thevline
1646         \let\tabu@celllalign \@empty
1647         \let\tabu@cellralign \@empty
1648         \let\tabu@cellrright \@empty
1649         \let\tabu@celllleft \@empty
1650         \let\tabu@thevline \relax}%
1651     \edef\tabu@temp{\tabu@multispan \tabu@nbc\cols{\noindent &}}%
1652     \toks@\expandafter{\tabu@temp \noindent\tabu@everyrowfalse \cr

```



```

1653      \noalign{\tabu@rearstrut
1654      {\globaldefs\@ne
1655      \let\tabu@celllalign \tabu@@celllalign
1656      \let\tabu@cellralign \tabu@@cellralign
1657      \let\tabu@cellleft \tabu@@cellleft
1658      \let\tabu@cellright \tabu@@cellright
1659      \let\tabu@thevline \tabu@@thevline}}}%
1660      \expandafter\the\toks@
1661 }% \tabuphantomline

```

## 11.16 Horizontal lines inside tabu: `\tabucline`, `\firsthline` and `\lasthline`

Horizontal lines: multiple `\firsthline` / `\lasthline`

`\tabu@firsthline` `\firsthline` and `\lasthline` are `\let` to `\tabu@firsthline` and `\tabu@lasthline` inside the `tabu` environment.

`\tabu@lasthline` This allows to duplicate horizontal lines, while keeping the alignment:

`\tabu@firsthline` `\firsthline \firsthline \firsthline` is allowed inside `tabu` and is the same as:

`\tabu@lasthline` `\firsthline \hline \hline`.

```

1662 \def\tabu@firsthline {\tabu@hlineAZ \tabu@firsthlinecorrection {}}
1663 \def\tabu@firsthline{\tabu@hlineAZ \tabu@firsthlinecorrection \hline}
1664 \def\tabu@lasthline {\tabu@hlineAZ \tabu@lasthlinecorrection {}}
1665 \def\tabu@lasthline {\tabu@hlineAZ \tabu@lasthlinecorrection \hline}
1666 \def\tabu@hline {% replaces \hline if no colortbl (see \AtBeginDocument)
1667     \noalign{\ifnum0='}\fi
1668     {\CT@arc@\hrule height\arrayrulewidth}%
1669     \futurelet \tabu@temp \tabu@xhline
1670 }% \tabu@hline
1671 \def\tabu@xhline{%
1672     \ifx \tabu@temp \hline
1673         {\ifx \CT@drsc@\relax \vskip
1674         \else\ifx \CT@drsc@\empty \vskip
1675         \else \CT@drsc@\hrule height
1676         \fi\fi
1677         \doublerulesep}%
1678     \fi
1679     \ifnum0='{ \fi}%
1680 }% \tabu@xhline

```

`\tabu@hlineAZ` Here we go, inside a `\noalign` group, we collect the next tokens:

- `\tabu@nexthlineAZ`
1. first the option,
  2. and then the next tokens if they are `\hline` or `\firsthline`.
- `\tabu@xhlineAZ`

The code to be executed at the end of the `\noalign` group is built into `\toks@`.

```

1681 \def\tabu@hlineAZ #1#2{\noalign{\ifnum0='}\fi \dimen@ \z@ \count@ \z@
1682     \toks@{}\def\tabu@hlinecorrection{#1}\def\tabu@temp{#2}%
1683     \tabu@hlineAZsurround
1684 }% \tabu@hlineAZ
1685 \newcommand*\tabu@hlineAZsurround[1][\extratabsurround]{%
1686     \extratabsurround #1\let\tabucline \tabucline@scan
1687     \let\hline \tabu@hlinescan \let\firsthline \hline
1688     \let\cline \tabu@clinescan \let\lasthline \hline
1689     \expandafter \futurelet \expandafter \tabu@temp
1690         \expandafter \tabu@nexthlineAZ \tabu@temp
1691 }% \tabu@hlineAZsurround
1692 \def\tabu@hlinescan {\tabu@thick \arrayrulewidth \tabu@xhlineAZ \hline}
1693 \def\tabu@clinescan #1{\tabu@thick \arrayrulewidth \tabu@xhlineAZ {\cline{#1}}}

```

```

1694 \def\tabucline@scan{\@testopt \tabucline@sc@n {}}
1695 \def\tabucline@sc@n #1[#2]{\tabu@xhlineAZ \tabucline[{#1}]{#2}}
1696 \def\tabu@nexthlineAZ{%
1697     \ifx \tabu@temp\hline \else
1698     \ifx \tabu@temp\cline \else
1699     \ifx \tabu@temp\tabucline \else
1700         \tabu@hlinecorrection
1701     \fi\fi\fi
1702 }% \tabu@nexthlineAZ
1703 \def\tabu@xhlineAZ #1{%
1704     \toks@{\expandafter{\the\toks@ #1}}
1705     \@tempdimc \tabu@thick % The last line width
1706     \ifcase\count@ \@tempdimb \tabu@thick % The first line width
1707     \else \advance\dimen@ \dimexpr \tabu@thick+\doublerulesep \relax
1708     \fi
1709     \advance\count@ \@ne \futurelet \tabu@temp \tabu@nexthlineAZ
1710 }% \tabu@xhlineAZ

```

**\tabu@firstlinecorrection** This is the “correction macro” for **\firstline**, *ie.*a strut and a skip are inserted **before** the first **\hline**.

```

1711 \def\tabu@firstlinecorrection{% \count@ = number of \hline -1
1712     \@tempdima \dimexpr \ht\@arstrutbox+\dimen@
1713     \edef\firstline{% <local in \noalign>
1714         \omit \hbox to\z@{\hss\{noexpand\tabu@DBG{yellow}\vrule
1715             height \the\dimexpr\@tempdima+\extratabsurround
1716             depth \dp\@arstrutbox
1717             width \tabustrutrule}\hss}\cr
1718         \noalign{\vskip -\the\dimexpr \@tempdima+\@tempdimb
1719             +\dp\@arstrutbox \relax}%
1720         \the\toks@
1721     }\ifnum0='\fi
1722     \expandafter\firstline % we are then !
1723 }% \tabu@firstlinecorrection

```

**\tabu@lastlinecorrection** This is the “correction macro” for **\lastline**, *ie.*a strut and a skip are inserted **after** the last **\hline**.

```

1724 \def\tabu@lastlinecorrection{%
1725     \@tempdima \dimexpr \dp\@arstrutbox+\dimen@+\@tempdimb+\@tempdimc
1726     \edef\lastline{% <local in \noalign>
1727         \the\toks@
1728         \noalign{\vskip-\the\dimexpr\dimen@+\@tempdimb+\dp\@arstrutbox}%
1729         \omit \hbox to\z@{\hss\{noexpand\tabu@DBG{yellow}\vrule
1730             depth \the\dimexpr \dp\@arstrutbox+\@tempdimb+\dimen@
1731                 +\extratabsurround-\@tempdimc
1732             height \z@
1733             width \tabustrutrule}\hss}\cr
1734     }\ifnum0='\fi
1735     \expandafter\lastline % we are then !
1736 }% \tabu@lastlinecorrection

```

## Horizontal lines: **\tabucline**

**\tabucline** **\tabucline** [style or spec.]{start-end}

**\tabucline** appears only at the end of a line: this is the place where we can insert a **\noalign** group. The line to be inserted inside the **tabu** is build inside this **\noalign** group.

**\tabu@start** and **\tabu@stop** store the limits for the line: they are, for clarity, the local name of **\@tempcnta** and **\@tempcntb**.

```

1737 \let\tabu@start \@tempcnta
1738 \let\tabu@stop \@tempcntb
1739 \newcommand*\tabucline{\noalign{\ifnum0='}\fi \tabu@ccline}
1740 \newcommand*\tabu@ccline[2][\tabu@startstop{#2}%
1741   \ifnum \tabu@stop<\z@ \toks@{}%
1742   \else \tabu@cclinearg{#1}\tabu@thestyle
1743     \edef\tabucline{\toks@{%
1744       \ifnum \tabu@start>\z@ \omit
1745         \tabu@multispan\tabu@start {\span\omit}&\fi
1746       \omit \tabu@multispan\tabu@stop {\span\omit}%
1747       \tabu@thehline\cr
1748     }}\tabucline
1749     \tabu@tracinglines{(tabu:tabucline) Style: #1^^J\the\toks@^^J^^J}%
1750   \fi
1751   \futurelet \tabu@temp \tabu@xccline
1752 }% \tabu@ccline
1753 \def\tabu@cclinearg #1{%
1754   \ifx\#1\\\let\tabu@thestyle \tabu@ls@
1755   \else \@defaultunits \expandafter\let\expandafter\@tempa
1756     \romannumeral-\0#1\relax \@nnil
1757     \ifx \hbox\@tempa \tabu@cclinebox{#1}%
1758     \else\ifx \box\@tempa \tabu@cclinebox{#1}%
1759     \else\ifx \vbox\@tempa \tabu@cclinebox{#1}%
1760     \else\ifx \vtop\@tempa \tabu@cclinebox{#1}%
1761     \else\ifx \copy\@tempa \tabu@cclinebox{#1}%
1762     \else\ifx \leaders\@tempa \tabu@cclineleads{#1}%
1763     \else\ifx \cleaders\@tempa \tabu@cclineleads{#1}%
1764     \else\ifx \xleaders\@tempa \tabu@cclineleads{#1}%
1765     \else\tabu@getline {#1}%
1766     \fi\fi\fi\fi\fi\fi\fi\fi
1767   \fi
1768 }% \tabu@cclinearg
1769 \def\tabu@cclinebox #1{\tabu@cclineleads{\xleaders#1\hss}}
1770 \def\tabu@cclineleads #1{%
1771   \let\tabu@thestyle \relax \let\tabu@leaders \@undefined
1772   \gdef\tabu@thehrule{#1}}
1773 \def\tabu@thehline{\begingroup
1774   \ifdefined\tabu@leaders
1775     \noexpand\tabu@thehleaders
1776   \else \noexpand\tabu@thehrule
1777   \fi \endgroup
1778 }% \tabu@thehline
1779 \def\tabu@xccline{%
1780   \ifx \tabu@temp\tabucline
1781     \toks@\expandafter{\the\toks@ \noalign
1782     {\ifx\CT@drsc@\relax \vskip
1783       \else \CT@drsc@\hrule height
1784       \fi
1785       \doublerulesep}}%
1786   \fi
1787   \tabu@docline
1788 }% \tabu@xccline
1789 \def\tabu@docline {\ifnum0='\fi \expandafter}\the\toks@}
1790 \def\tabu@docline@evr {\xdef\tabu@doclineafter{\the\toks@}%
1791   \ifnum0='\fi}\aftergroup\tabu@doclineafter}
1792 \def\tabu@multispan #1#2{%
1793   \ifnum\numexpr#1>\@ne #2\expandafter\tabu@multispan
1794   \else \expandafter\@gobbletwo

```

```

1795      \fi    {#1-1}{#2}%
1796 }% \tabu@multispan

```

**\tabu@startstop** This macro parses the mandatory argument of `\tabucline`: start-column and end-column of the `\cline`.

```

1797 \def\tabu@start@stop #1{\tabu@start@stop #1\relax 1-\tabu@nbcols \@nnil}
1798 \def\tabu@start@stop #1-#2\@nnil{%
1799     \@defaultunits \tabu@start\number 0#1\relax \@nnil
1800     \@defaultunits \tabu@stop \number 0#2\relax \@nnil
1801     \tabu@stop \ifnum \tabu@start>\tabu@nbcols \m@ne
1802                 \else\ifnum \tabu@stop=\z@ \tabu@nbcols
1803                 \else\ifnum \tabu@stop>\tabu@nbcols \tabu@nbcols
1804                 \else \tabu@stop
1805                 \fi\fi\fi
1806     \advance\tabu@start \m@ne
1807     \ifnum \tabu@start>\z@ \advance\tabu@stop -\tabu@start \fi
1808 }% \tabu@start@stop

```

### 11.17 Numbers in tabu

## \tabudecimal

```
\tabudecimal \tabu@tabudecimal is \tabudecimal inside the tabu environment.
```

```

1809 \def\tabu@tabudecimal #1{%
1810     \def\tabu@decimal{#1}\@temptokena{}%
1811     \let\tabu@getdecimal@ \tabu@getdecimal@ignorespaces
1812     \tabu@scandecimal
1813 }% \tabu@tabudecimal
1814 \def\tabu@scandecimal{\futurelet \tabu@temp \tabu@getdecimal@}
1815 \def\tabu@skipdecimal#1{#1\tabu@scandecimal}
1816 \def\tabu@getdecimal@ignorespaces{%
1817     \ifcase 0\ifx\tabu@temp\ignorespaces\else
1818         \ifx\tabu@temp\@sptoken1\else
1819             2\fi\fi\relax
1820     \let\tabu@getdecimal@ \tabu@getdecimal
1821     \expandafter\tabu@skipdecimal
1822 \or \expandafter\tabu@gobblespace\expandafter\tabu@scandecimal
1823 \else \expandafter\tabu@skipdecimal
1824 \fi
1825 }% \tabu@getdecimal@ignorespaces
1826 \def\tabu@get@decimal#1{\@temptokena\expandafter{\the\@temptokena #1}%
1827     \tabu@scandecimal}
1828 \def\do#1{%
1829     \def\tabu@get@decimalspace#1{%
1830         \@temptokena\expandafter{\the\@temptokena #1}\tabu@scandecimal}%
1831 } \do{ }
1832 \let\tabu@@@tabudecimal \tabu@tabudecimal

```

\tabu@getdecimal

```

1833 \def\tabu@getdecimal{%
1834     \ifcase      0\ifx 0\tabu@temp\else
1835         \ifx 1\tabu@temp\else
1836         \ifx 2\tabu@temp\else
1837         \ifx 3\tabu@temp\else
1838         \ifx 4\tabu@temp\else
1839         \ifx 5\tabu@temp\else
1840         \ifx 6\tabu@temp\else
1841         \ifx 7\tabu@temp\else
1842         \ifx 8\tabu@temp\else

```

```

1843         \ifx 9\tabu@temp\else
1844         \ifx .\tabu@temp\else
1845         \ifx ,\tabu@temp\else
1846         \ifx -\tabu@temp\else
1847         \ifx +\tabu@temp\else
1848         \ifx e\tabu@temp\else
1849         \ifx E\tabu@temp\else
1850         \ifx\tabu@celleft\tabu@templ\else
1851         \ifx\ignorespaces\tabu@templ\else
1852         \ifx\@sptoken\tabu@temp2\else
1853         3\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\fi\relax
1854     \expandafter\tabu@get@decimal
1855 \or \expandafter\tabu@skipdecimal
1856 \or \expandafter\tabu@get@decimalspace
1857 \else\expandafter\tabu@printdecimal
1858 \fi
1859 }% \tabu@getdecimal
1860 \def\tabu@printdecimal{%
1861     \edef\tabu@temp{\the\@temptokena}%
1862     \ifx\tabu@temp\@empty\else
1863     \ifx\tabu@temp\space\else
1864         \expandafter\tabu@decimal\expandafter{\the\@temptokena}%
1865     \fi\fi
1866 }% \tabu@printdecimal

```

## 11.18 Verbatim inside tabu with X columns

**\tabu@verbatim** Setup to be done before `\scantokens` to allow verbatim inside the `tabu` environment.

```

1867 \def\tabu@verbatim{%
1868     \let\verb \tabu@verb
1869     \let\FV@DefineCheckEnd \tabu@FV@DefineCheckEnd
1870 }% \tabu@verbatim

```

### Compatibility with L<sup>A</sup>T<sub>E</sub>X's kernel `\verb` command

**\tabu@verb** The `\verb` macro from the latex kernel expands `\@ifstar` in a context where the space token: `\`  has a category code of 12.

This is not compatible with `\scantokens` since `\scantokens` adds a space after each control sequence, including `\verb`:

`\verb +some verbatim text+` becomes:

`\verb \ +some verbatim text+`

and thus, the space token `\`  is set as the `\verb` delimiter.

We therefore use (a silly) `\@ifstar` in order to gobble the possible space token.

```

1871 \let\tabu@ltx@verb \verb
1872 \def\tabu@verb{\@ifstar {\tabu@ltx@verb*} \tabu@ltx@verb}

```

### Compatibility with the `fancyvrb` package

**\tabu@FV@DefineCheckEnd** This is quite the same issue as for L<sup>A</sup>T<sub>E</sub>X `\verb` command: a space is inserted after each control sequence scanned by `\scantoken`.

This leads to a break in the macro that checks the end of a `Verbatim` environment, since this macro basically checks for a line that conforms to the pattern:

**#1\end {#2}#3**

while with `\scantokens`, such a line becomes:

**#1\end \{#2}#3**

in a context where the space token is not of category 10 (space).

Thus we replace the end-check for the `Verbatim` environment by a check on the detokenized-line (with `\detokenize`):

```

1873 \def\tabu@fancyvrb {%
1874     \def\tabu@FV@DefineCheckEnd ##1{%
1875         \def\tabu@FV@DefineCheckEnd{%
1876             ##1% <original definition (if fancyvrb is loaded)>
1877             \let\FV@CheckEnd      \tabu@FV@CheckEnd
1878             \let\FV@@CheckEnd     \tabu@FV@@CheckEnd
1879             \let\FV@@@CheckEnd    \tabu@FV@@@CheckEnd
1880             \edef\FV@EndScanning{%
1881                 \def\noexpand\next{\noexpand\end{\FV@EnvironName}}%
1882                 \global\let\noexpand\FV@EnvironName\relax
1883                 \noexpand\next}%
1884             \xdef\FV@EnvironName{\detokenize\expandafter{\FV@EnvironName}}}%
1885         }\expandafter\tabu@FV@DefineCheckEnd\expandafter{\FV@DefineCheckEnd}
1886     }% \tabu@fancyvrb
1887 \def\tabu@FV@CheckEnd  #1{\expandafter\FV@@CheckEnd \detokenize{#1\end{}}\@nil}
1888 \edef\tabu@FV@@@CheckEnd {\detokenize{\end{}}}
1889 \begingroup
1890 \catcode`\[1      \catcode`\]2
1891 \@makeoother{\    \@makeoother\}
1892     \edef\x[\endgroup
1893         \def\noexpand\tabu@FV@@@CheckEnd ##1\detokenize[\end{}}##2\detokenize[]##3%
1894     ]\x          \@nil{\def\@tempa{#2}\def\@tempb{#3}}

```

**\tabu@FV@ListProcessLine** This macro replaces `\FV@ListProcessLine` when measuring the natural width of a `Verbatim` environment (see `\tabu@startpboxmeasure`)

```

1895 \def\tabu@FV@ListProcessLine #1{%
1896     \hbox {%to \hsize{%
1897         \kern\leftmargin
1898         \hbox {%to \linewidth{%
1899             \FV@LeftListNumber
1900             \FV@LeftListFrame
1901             \FancyVerbFormatLine{#1}\hss
1902 %% DG/SR modification begin - Jan. 28, 1998 (for numbers=right add-on)
1903 %%         \FV@RightListFrame}%
1904             \FV@RightListFrame
1905             \FV@RightListNumber}%
1906 %% DG/SR modification end
1907         \hss}}

```

## 11.19 \savetabu

**\savetabu** When this command is called by the user, the `tabu` preamble and target are globally stored into a macro `\tabu@saved@{user-name}`.

```

1908 \newcommand*\savetabu[1]{\noalign{%
1909     \tabu@sanitizearg{#1}\tabu@temp
1910     \ifx \tabu@temp\empty \tabu@savewarn{}{The tabu will not be saved}\else
1911         \@ifundefined{tabu@saved@\tabu@temp}{}{\tabu@savewarn{#1}{Overwriting}}%
1912         \ifdefined\tabu@restored \expandafter\let
1913             \csname tabu@saved@\tabu@temp \endcsname \tabu@restored
1914         \else {\tabu@save}%
1915     \fi
1916     \fi}%
1917 }% \savetabu
1918 \def\tabu@save {%

```

```

1919 \toks0\expandafter{\tabu@savewd}%
1920 \iftabu@negcoef
1921 \let\tabu@wddef \relax \let\tabu@ \tabu@savewd \edef\tabu@savewd{\tabu@Xcoefs}%
1922 \toks0\expandafter{\the\toks\expandafter0\tabu@savewd}\fi
1923 \toks1\expandafter{\tabu@savewd}%
1924 \toks2\expandafter{\tabu@savewd}%
1925 \let\@preamble \relax
1926 \let\tabu@savewd \relax \let\tabu@savewdparams \relax
1927 \edef\tabu@preamble{%
1928 \def\noexpand\tabu@aligndefault{\tabu@align}%
1929 \def\tabu@savewdparams {\noexpand\the\toks0}%
1930 \def\tabu@savewdpreamb {\noexpand\the\toks1}}%
1931 \edef\tabu@usetabu{%
1932 \def\@preamble {\noexpand\the\toks2}%
1933 \tabu@target \the\tabu@target \relax
1934 \tabu@colX \the\tabu@colX \relax
1935 \tabu@nbcols \the\tabu@nbcols \relax
1936 \def\noexpand\tabu@aligndefault{\tabu@align}%
1937 \def\tabu@savewdparams {\noexpand\the\toks0}%
1938 \def\tabu@savewdpreamb {\noexpand\the\toks1}}%
1939 \let\tabu@aligndefault \relax \let\@sharp \relax
1940 \edef\@tempa{\noexpand\tabu@s@ved
1941 \tabu@usetabu}
1942 \tabu@preamble
1943 \the\toks1}}\@tempa
1944 \tabu@message@save
1945 }% \tabu@save
1946 \long\def\tabu@s@ved #1#2#3{%
1947 \def\tabu@usetabu{#1}% <for \tabu@message@save>
1948 \expandafter\gdef\csname tabu@savewd\tabu@temp\endcsname ##1{%
1949 \ifodd ##1% \usetabu
1950 \tabu@measuringfalse \tabu@spreadfalse % Just in case...
1951 \gdef\tabu@usetabu {%
1952 \ifdim \tabu@target>\z@ \tabu@warn@usetabu \fi
1953 \global\let\tabu@usetabu \@undefined
1954 \def\@halignto {to\tabu@target}%
1955 #1%
1956 \ifx \tabu@align\tabu@aligndefault@text
1957 \ifnum \tabu@nested=\z@
1958 \let\tabu@align \tabu@aligndefault \fi\fi}%
1959 \else % \preamble
1960 \gdef\tabu@preamble {%
1961 \global\let\tabu@preamble \@undefined
1962 #2%
1963 \ifx \tabu@align\tabu@aligndefault@text
1964 \ifnum \tabu@nested=\z@
1965 \let\tabu@align \tabu@aligndefault \fi\fi}%
1966 \fi
1967 #3}%
1968 }% \tabu@s@ved
1969 \def\tabu@aligndefault@text {\tabu@aligndefault}%
1970 \def\tabu@warn@usetabu {\PackageWarning{tabu}
1971 {Specifying a target with \string\usetabu\space is useless
1972 \MessageBreak The target cannot be changed!}}
1973 \def\tabu@savewd #1#2{\ifdim #2\p@<\z@ \tabu@wddef{#1}{\tabu@wd{#1}}\fi}

```



Error if `\usetabu` is called with an unknown argument.

```

1974 \def\tabu@savewarn#1#2{\PackageInfo{tabu}
1975     {User-name '#1' already used for \string\savetabu
1976     \MessageBreak #2}}%
1977 \def\tabu@saveerr#1{\PackageError{tabu}
1978     {User-name '#1' is unknown for \string\usetabu
1979     \MessageBreak I cannot restore an unknown preamble!}\@ehd}

```

## 11.20 `\rowfont`

### Setting font and alignment specification

**`\rowfont`** `\rowfont` uses the control sequences `\tabu@celllalign`, `\tabu@celleft`, `\tabu@cellright` and `\tabu@cellralign` which have been placed on purpose into the user-defined tokens inserted in any preamble by the array package.

`\tabu@celllalign` and `\tabu@cellralign` are used to modify the alignment. If the optional [alignment] parameter of `\rowfont` is not specified, then those control sequences expand to `\@empty`.

`\tabu@celleft` contains the font-modification information.

Placement of those control sequences into the user-tokens that are inserted in the preamble by the array package is explained below under the macro `\tabu@prepnext@tok`.

```

1980 \newskip \tabu@cellskip
1981 \def\tabu@rowfont{\ifdim \baselineskip=\z@\noalign\fi
1982     {\ifnum0='}\fi \tabu@row@font}
1983 \newcommand*\tabu@row@font[2][{}]{%
1984     \ifnum7=\currentgrouptype
1985         \global\let\tabu@@celleft \tabu@celleft
1986         \global\let\tabu@@cellright \tabu@cellright
1987         \global\let\tabu@@celllalign \tabu@celllalign
1988         \global\let\tabu@@cellralign \tabu@cellralign
1989         \global\let\tabu@@rowfontreset\tabu@rowfontreset
1990     \fi
1991     \global\let\tabu@rowfontreset \tabu@rowfont@reset
1992     \expandafter\gdef\expandafter\tabu@celleft\expandafter{\tabu@celleft #2}%
1993     \ifcsname tabu@cell@#1\endcsname % row alignment
1994         \csname tabu@cell@#1\endcsname \fi
1995     \ifnum0='{'\fi}% end of group / noalign group
1996 }% \rowfont
1997 \def\tabu@ifcolorleavevmode #1{\let\color \tabu@leavevmodecolor #1\let\color\tabu@color}%

```

**`\tabu@rowfont@reset`** This macro restores `\tabu@celllalign`, `\tabu@celleft`, `\tabu@cellright`, and `\tabu@cellralign` to the value they had before the expansion of `\rowfont`.

It expands when a new row is inserted into the tabular or array.

```

1998 \def\tabu@rowfont@reset{%
1999     \global\let\tabu@rowfontreset \tabu@@rowfontreset
2000     \global\let\tabu@celleft \tabu@@celleft
2001     \global\let\tabu@cellright \tabu@@cellright
2002     \global\let\tabu@cellfont \@empty
2003     \global\let\tabu@celllalign \tabu@@celllalign
2004     \global\let\tabu@cellralign \tabu@@cellralign
2005 }% \tabu@@rowfontreset
2006 \let\tabu@rowfontreset \@empty % overwritten \AtBeginDocument if colortbl

```

## Preparing stuff to be able to use `\rowfont`

`\tabu@prepnext@tok` `\tabu@prepnext@tok` will replace `\prepnext@tok` defined in `array.sty` (only inside a `tabu` environment). its purpose is to count the number of columns, and to insert the control sequences `\tabu@cellalign`, `\tabu@celleft`, `\tabu@cellright` and `\tabu@cellralign` at the edge of each cell of the tabular. This is done by putting them inside the user-tokens placed around each column by the `array` package.

`\prepnext@tok` in `array.sty` initialises each user-token to an empty one, each time there is a need for a new one ! The macro has a very simple definition, but its expansion is the occasion to look carefully at the counters `\count@` and `\@tempcnta` which gives us all information required to decide if the token in preparation will be finally placed on the left or on the right of a column.

$$\underbrace{\langle \{\texttt{\textbackslash bfseries} \texttt{\textbackslash color \{red\}} \} \rangle}_{\texttt{\textbackslash toks} < i >} \text{ r } \underbrace{\langle \{\texttt{\textbackslash color \{black\}} \texttt{\textbackslash}, \texttt{\textbackslash $} \} \rangle}_{\texttt{\textbackslash toks} < i + 1 >}$$

When a column is inserted in the tabular preamble (`\@preamble`), the T<sub>E</sub>X counter `\count@` is equal to  $i + 1$  (*ie.* the right token) and the counter `\@tempcnta` is equal to  $i$  (*ie.* the left token). If the column is special (*ie.* `@` or `!`) `\@tempcnta` is not updated.

Thus, when a new token is “prepared” by `\prepnext@tok`:

**either:** `i = \count@ = \@tempcnta` : the token to prepare (*ie.* `\toks < i + 1 >`) is the right one of a “normal” column. The switch `\iftabu@cellright` is set to `true`.

The *previous* token (`\toks < i > = \toks \count@`) is necessarily the left one of this “normal” column: we prepend `\tabu@cellalign` and append `\tabu@celleft` to this token (`\toks < i >`). This token is finished and will not change afterwards.

**or:** `i = \count@ = \@tempcnta + 1` : the token to prepare (`\toks < i + 1 >`) is either the left one of a normal column, or the single one of a special `@` or `!` column.

If the switch `\iftabu@cellright` is `true`, then the *previous* token `\toks < i >` is the right one of the last inserted column (which was a “normal” column, thus); `\tabu@cellright \tabu@cellralign` is appended to it, and the switch `\iftabu@cellright` is reset to `false`. May be `\prepnext@tok` will be expanded again (by `\save@decl`): if it happens, then again `\count@ = \@tempcnta + 1` (same case) but `\iftabu@cellright` is `false` and nothing is changed.

**else:** The token to prepare (which is `\toks < i + 1 > = \toks \count@ + 1`), cannot be the right one of a “normal” column: `\iftabu@cellright` is set to `false`.

The fact that  $|\texttt{\textbackslash count@} - \texttt{\@tempcnta}| > 1$  tells us that the previous token `\toks < i >` is necessarily the single one of a “special” `@` or `!` column. We don’t modify this token, as long as *special columns are always inserted as is*: `\rowcolor` has no effect on special columns, nor `\rowfont`.

Thereafter, the original initialisation sequence occurs: `\advance \count@ by \@one` and initialize the token to prepare (`\toks \count@ = \toks < i + 1 >`) to an empty one.

```
2007 \newif \iftabu@cellright
2008 \def\tabu@prepnext@tok{%
2009     \ifnum \count@<\z@    % <first initialisation>
2010         \@tempcnta \@M    % <not initialized by array.sty>
2011         \tabu@nbcols\z@
2012         \let\tabu@fornoopORI \@fornoop
2013         \tabu@cellrightfalse
2014     \else
2015         \ifcase \numexpr \count@-\@tempcnta \relax % (case 0): prev. token is left
2016             \advance \tabu@nbcols \@ne
2017             \iftabu@cellright % before-previous token is right and is finished
2018                 \tabu@cellrightfalse % <only once>
2019                 \tabu@righttok
2020             \fi
2021             \tabu@lefttok
2022         \or % (case 1) previous token is right
2023             \tabu@cellrighttrue \let\tabu@fornoop \tabu@lastnoop
2024         \else % special column: do not change the token
2025             \iftabu@cellright % before-previous token is right
```

```

2026          \tabu@cellrightfalse
2027          \tabu@righttok
2028          \fi
2029          \fi % \ifcase
2030          \fi
2031          \tabu@prepnext@tokORI
2032}% \tabu@prepnext@tok
2033\long\def\tabu@lastnoop#1\@@#2#3{\tabu@lastn@@p #2\@nextchar \in@\in@@}
2034\def\tabu@lastn@@p #1\@nextchar #2#3\in@@{%
2035    \ifx \in@#2\else
2036      \let\@fornoop \tabu@fornoopORI
2037      \xdef\tabu@mkpreamblebuffer{\tabu@nbc\cols\the\tabu@nbc\cols \tabu@mkpreamblebuffer}%
2038      \toks0\expandafter{\expandafter\tabu@everyrowtrue \the\toks0}%
2039      \expandafter\prepnext@tok
2040    \fi
2041}% \tabu@lastnoop
2042\def\tabu@righttok{%
2043  \advance \count@ \m@ne
2044  \toks\count@\expandafter {\the\toks\count@ \tabu@cellright \tabu@cellralign}%
2045  \advance \count@ \@ne
2046}% \tabu@righttok
2047\def\tabu@lefttok{\toks\count@\expandafter{\expandafter\tabu@celllalign
2048                                     \the\toks\count@ \tabu@cellleft}% after because of $
2049}% \tabu@lefttok

```

## Neutralisation of glues and alignment modification

**`\tabu@cellleft`** First initialisation to `\@empty`.

```

\tabu@celllalign 2050 \let\tabu@cellleft \@empty
\tabu@cellright 2051 \let\tabu@cellright \@empty
\tabu@cellralign 2052 \tabu@celllalign\def{\tabu@cellleft}%
2053 \let\tabu@cellralign \@empty

```

**`\tabu@cell@align`**

```

2054 \def\tabu@cell@align #1#2#3{%
2055   \let\tabu@maybesiunitx \toks@ \tabu@celllalign
2056   \global \expandafter \tabu@celllalign\def \expandafter {\the\toks@ #1}%
2057   \toks@\expandafter{\tabu@cellralign #2}%
2058   \xdef\tabu@cellralign{\the\toks@}%
2059   \toks@\expandafter{\tabu@cellleft #3}%
2060   \xdef\tabu@cellleft{\the\toks@}%
2061}% \tabu@cell@align

```

**`\tabu@cell@l`** Setup macros to modify the alignment. The skips inserted to make the standard alignment specified in the tabular preamble are not the same with standard `array` tabulars and `colortbl` tabulars, hence the switch `\iftabu@colortbl`.

```

\tabu@cell@c 2062 \def\tabu@cell@l{% force alignment to left
\tabu@cell@r 2063   \tabu@cell@align
\tabu@cell@j 2064     {\tabu@removehfil \raggedright \tabu@cellleft}% left
2065     {\tabu@flush1\tabu@ignorehfil}% right
2066     \raggedright
2067}% \tabu@cell@l
2068 \def\tabu@cell@c{% force alignment to center
2069   \tabu@cell@align
2070     {\tabu@removehfil \centering \tabu@flush{.5}\tabu@cellleft}
2071     {\tabu@flush{.5}\tabu@ignorehfil}
2072     \centering
2073}% \tabu@cell@c

```

```

2074 \def\tabu@cell@r{% force alignment to right
2075     \tabu@cell@align
2076     {\tabu@removehfil \raggedleft \tabu@flush1\tabu@cellleft}
2077     \tabu@ignorehfil
2078     \raggedleft
2079 }% \tabu@cell@r
2080 \def\tabu@cell@j{% force justification (for p, m, b columns)
2081     \tabu@cell@align
2082     {\tabu@justify\tabu@cellleft}
2083     {}
2084     \tabu@justify
2085 }% \tabu@cell@j
2086 \def\tabu@justify{%
2087     \leftskip\z@skip \@rightskip\leftskip \rightskip\@rightskip
2088     \parfillskip\@flushglue
2089 }% \tabu@justify
2090 %% ragged2e settings
2091 \def\tabu@cell@L{% force alignment to left (ragged2e)
2092     \tabu@cell@align
2093     {\tabu@removehfil \RaggedRight \tabu@cellleft}
2094     {\tabu@flush 1\tabu@ignorehfil}
2095     \RaggedRight
2096 }% \tabu@cell@L
2097 \def\tabu@cell@C{% force alignment to center (ragged2e)
2098     \tabu@cell@align
2099     {\tabu@removehfil \Centering \tabu@flush{.5}\tabu@cellleft}
2100     {\tabu@flush{.5}\tabu@ignorehfil}
2101     \Centering
2102 }% \tabu@cell@C
2103 \def\tabu@cell@R{% force alignment to right (ragged2e)
2104     \tabu@cell@align
2105     {\tabu@removehfil \RaggedLeft \tabu@flush 1\tabu@cellleft}
2106     \tabu@ignorehfil
2107     \RaggedLeft
2108 }% \tabu@cell@R
2109 \def\tabu@cell@J{% force justification (ragged2e)
2110     \tabu@cell@align
2111     {\justifying \tabu@cellleft}
2112     {}
2113     \justifying
2114 }% \tabu@cell@J
2115 \def\tabu@flush#1{%
2116     \iftabu@colortbl      % colortbl uses \hfill rather than \hfil
2117         \hskip \ifnum13<\currentgroup\type \stretch{#1}%
2118     \else \ifdim#1pt<\p@ \tabu@cellskip
2119     \else \stretch{#1}
2120     \fi\fi \relax
2121 \else
2122     \ifnum 13<\currentgroup\type
2123         \hfil \hskip1sp \relax \fi
2124 \fi
2125 }% \tabu@flush

```

**\tabu@removehfil** **\tabu@removehfil** removes (eventually) the infinite stretchable glue inserted *before* the cell (in the preamble of **\halign**) to make the column alignment.

```

2126 \let\tabu@hfil \hfil
2127 \let\tabu@hfill \hfill

```

```

2128 \let\tabu@hskip \hskip
2129 \def\tabu@removehfil{%
2130     \iftabu@colortbl
2131         \unkern \tabu@cellskip =\lastskip
2132         \ifnum\gluestretchorder\tabu@cellskip =\tw@ \hskip-\tabu@cellskip
2133         \else \tabu@cellskip \z@skip
2134         \fi
2135     \else
2136         \ifdim\lastskip=1sp\unskip\fi
2137         \ifnum\gluestretchorder\lastskip =\@ne
2138             \hfilneg % \hfilneg for array.sty but not for colortbl...
2139         \fi
2140     \fi
2141 }% \tabu@removehfil

```

**\tabu@ignorehfil** **\tabu@ignorehfil** removes (eventually) the infinite stretchable glue inserted *after* the cell (in the preamble of **\halign**) to make the column alignment.

```

2142 \def\tabu@ignorehfil{\aftergroup \tabu@nohfil}
2143 \def\tabu@nohfil{% \hfil -> do nothing + restore original \hfil
2144     \def\hfil{\let\hfil \tabu@hfil}% local to (alignment template) group
2145 }% \tabu@nohfil
2146 \def\tabu@colortblalignments {% if colortbl
2147     \def\tabu@nohfil{%
2148         \def\hfil {\let\hfil \tabu@hfil}% local to (alignment template) group
2149         \def\hfill {\let\hfill \tabu@hfill}% (colortbl uses \hfill) pfff...
2150         \def\hskip ###1\relax{\let\hskip \tabu@hskip}}% local
2151 }% \tabu@colortblalignments

```

## 11.21 Taking care of footnotes and \arraybackslash

### Footnotes and hyperfootnotes

**\tabu@footnotetext** The macros in case **hyperref** is not used, or used with the option **hyperfootnotes=false**:

```

2152 \long\def\tabu@footnotetext #1{%
2153     \edef\@tempa{\the\tabu@footnotes
2154         \noexpand\footnotetext [\the\csname c@\@mpfn\endcsname]}%
2155     \global\tabu@footnotes\expandafter{\@tempa {#1}}}%
2156 \long\def\tabu@xfootnotetext [#1]#2{%
2157     \global\tabu@footnotes\expandafter{\the\tabu@footnotes
2158         \footnotetext [{#1}]{#2}}
2159 \let\tabu@xfootnote \@xfootnote

```

**\tabu@Hy@ftntext** The macros in case **hyperref** is loaded with the option **hyperfootnotes=true**:

```

\tabu@Hy@xfootnote 2160 \long\def\tabu@Hy@ftntext{\tabu@Hy@ftntxt {\the \c@footnote }}
2161 \long\def\tabu@Hy@xfootnote [#1]{%
2162     \begingroup
2163         \value\@mpfn #1\relax
2164         \protected@xdef \@thefnmark {\thempfn}%
2165     \endgroup
2166     \@footnotemark \tabu@Hy@ftntxt {#1}%
2167 }% \tabu@Hy@xfootnote
2168 \long\def\tabu@Hy@ftntxt #1#2{%
2169     \edef\@tempa{%
2170         \the\tabu@footnotes
2171         \begingroup
2172             \value\@mpfn #1\relax
2173             \noexpand\protected@xdef\noexpand\@thefnmark {\noexpand\thempfn}%
2174             \expandafter \noexpand \expandafter

```

```

2175          \tabu@Hy@footnotetext \expandafter{\Hy@footnote@currentHref}%
2176      }%
2177      \global\tabu@footnotes\expandafter{\@tempa {#2}%
2178          \endgroup}%
2179 }% \tabu@Hy@ftntxt
2180 \long\def\tabu@Hy@footnotetext #1#2{%
2181     \H@@footnotetext{%
2182         \ifHy@nesting
2183             \hyper@@anchor {#1}{#2}%
2184         \else
2185             \Hy@raisedlink{%
2186                 \hyper@@anchor {#1}{\relax}%
2187             }%
2188             \def\@currentHref {#1}%
2189             \let\@currentlabelname \@empty
2190             #2%
2191         \fi
2192     }%
2193 }% \tabu@Hy@footnotetext

```

## \centering, \raggedright, \raggedleft and \@normalcr

Inside `tabu` environment, no need to add `\arraybackslash` after such commands.

```

2194 \def\tabu@latextwoe {%
2195 \def\tabu@temp##1##2##3{{\toks@\expandafter{##2##3}\xdef##1{\the\toks@}}}
2196 \tabu@temp \tabu@centering \centering \arraybackslash
2197 \tabu@temp \tabu@raggedleft \raggedleft \arraybackslash
2198 \tabu@temp \tabu@raggedright \raggedright \arraybackslash
2199 }% \tabu@latextwoe
2200 \def\tabu@raggedtwoe {%
2201 \def\tabu@temp ##1##2##3{{\toks@\expandafter{##2##3}\xdef##1{\the\toks@}}}
2202 \tabu@temp \tabu@Centering \Centering \arraybackslash
2203 \tabu@temp \tabu@RaggedLeft \RaggedLeft \arraybackslash
2204 \tabu@temp \tabu@RaggedRight \RaggedRight \arraybackslash
2205 \tabu@temp \tabu@justifying \justifying \arraybackslash
2206 }% \tabu@raggedtwoe
2207 \def\tabu@normalcrbackslash{\let\\\@normalcr}
2208 \def\tabu@trivlist{\expandafter\def\expandafter\@trivlist\expandafter{%
2209     \expandafter\tabu@normalcrbackslash \@trivlist}}

```

## Utilities: tabu \fbox

`\tabu@fbox` works exactly like L<sup>A</sup>T<sub>E</sub>X `\fbox` but allows the syntax: `\fbox \bgroup...\egroup` suitable for use inside tabular columns. `\fbox` is `\let` to `\tabu@fbox` at the entry inside a `tabu` environment.

```

2210 \def\tabu@fbox      {\leavevmode\afterassignment\tabu@beginfbox \setbox\@tempboxa\hbox}
2211 \def\tabu@beginfbox {\bgroup \kern\fboxsep
2212                     \bgroup\aftergroup\tabu@endfbox}
2213 \def\tabu@endfbox   {\kern\fboxsep\egroup\egroup
2214                     \@frameb@x\relax}

```

`\tabu@fcolorbox` works exactly like xcolor `\fcolorbox` but allows the syntax:

`\fcolorbox {frame color}{background color}\bgroup...\egroup`

suitable for use insed tabular columns. `\fcolorbox` is `\let` to `\tabu@fcolorbox` at the entry inside a `tabu` environment.

```

2215 \def\tabu@color@b@x #1#2{\leavevmode \bgroup
2216     \def\tabu@docolor@b@x{#1{#2\color@block{\wd\z@}{\ht\z@}{\dp\z@}\box\z@}}%
2217     \afterassignment\tabu@begincolor@b@x \setbox\z@ \hbox

```

```

2218 }% \tabu@color@b@x
2219 \def\tabu@begincolor@b@x {\kern\fboxsep \bgroup
2220     \aftergroup\tabu@endcolor@b@x \set@color}
2221 \def\tabu@endcolor@b@x {\kern\fboxsep \egroup
2222     \dimen@ht\z@ \advance\dimen@ \fboxsep \ht\z@ \dimen@
2223     \dimen@dp\z@ \advance\dimen@ \fboxsep \dp\z@ \dimen@
2224     \tabu@docolor@b@x \egroup
2225 }% \tabu@endcolor@b@x

```

## 11.22 Corrections

### delarray comptability fix for colortbl and arydshln

Both colortbl and arydshln forgot the control sequence `\@arrayright` which must be expanded by `\endarray`. Originally defined for delarray, this control sequence is used by tabu environments when tabu X columns are present in the preamble.

Here is the fix. We test if `\endarray` contains `\@arrayright` before modifying the control sequence, in case colortbl and/or arydshln modify their definition.

```

2226 \def\tabu@fix@arrayright {% \@arrayright is missing from \endarray
2227     \iftabu@colortbl
2228         \ifdefined\adl@array % <colortbl + arydshln>
2229         \def\tabu@endarray{%
2230             \adl@endarray \egroup \adl@arrayrestore \CT@end \egroup %<original>
2231             \@arrayright % <FC>
2232             \gdef\@preamble{}}% <FC>
2233         \else % <colortbl / no arydshln>
2234         \def\tabu@endarray{%
2235             \crrc \egroup \egroup %<original>
2236             \@arrayright % <FC>
2237             \gdef\@preamble{\CT@end}%
2238         \fi
2239     \else
2240         \ifdefined\adl@array % <arydshln / no colortbl>
2241         \def\tabu@endarray{%
2242             \adl@endarray \egroup \adl@arrayrestore \egroup %<original>
2243             \@arrayright % <FC>
2244             \gdef\@preamble{}}% <FC>
2245         \else % <no arydshln / no colortbl + \@arrayright missing>
2246             \PackageWarning{tabu}
2247             {\string\@arrayright\space is missing from the
2248             \MessageBreak definition of \string\endarray.
2249             \MessageBreak Comptability with delarray.sty is broken.}%
2250         \fi\fi
2251 }% \tabu@fix@arrayright

```

### arydshln @ columns

```

2252 \def\tabu@adl@xarraydashrule #1#2#3{%
2253     \ifnum\@lastchclass=\adl@class@start\else
2254     \ifnum\@lastchclass=\@ne\else
2255     \ifnum\@lastchclass=5 \else % <FC> @-arg (class 5) and !-arg (class 1)
2256         \adl@leftrulefalse \fi\fi % must be treated the same
2257     \fi
2258     \ifadl@zwvrule\else \ifadl@inactive\else
2259         \@addtopreamble{\vrule\@width\arrayrulewidth
2260             \@height\z@ \@depth\z@}\fi \fi
2261     \ifadl@leftrule
2262         \@addtopreamble{\adl@vlineL{\CT@arc@}{\adl@dashgapcolor}%

```



```

2263             {\number#1}\#3}%
2264     \else     \@addtopreamble{\adl@vlineR{\CT@arc@}{\adl@dashgapcolor}%
2265             {\number#2}\#3}%
2266     \fi
2267 }% \tabu@adl@xarraydashrule

```

## arydshln, colors without colortbl and empty p columns

arydshln redefines \@endpbox for p columns. The definition is stored in \adl@act@endpbox. Here it is:

```

\unskip \ifhmode \nobreak
  \vrule\@width\z@\@height\z@\@depth\dp\@arstrutbox
\fi
\egroup \adl@colhtdp \box\adl@box \hfil

```

The \vrule inserted is exactly what package array calls: \@finalstrut \@arstrutbox.

However, just like in array.sty, this array-strut should be inserted inconditionnally, and \ifhmode applies only to \nobreak (misplaced \fi in arydshln definition).

Finally, arydshln is not compatible with colors in columns, such that: >{\color {red}}p3in, Unless colortbl is also loaded, the color group is missing.

Fixed inside tabu environment.

```

2268 \def\tabu@adl@act@endpbox {%
2269     \unskip \ifhmode \nobreak \fi     \@finalstrut \@arstrutbox
2270     \egroup \egroup
2271     \adl@colhtdp \box\adl@box \hfil
2272 }% \tabu@adl@act@endpbox
2273 \def\tabu@adl@fix {%
2274     \let\adl@xarraydashrule \tabu@adl@xarraydashrule % <fix> arydshln
2275     \let\adl@act@endpbox     \tabu@adl@act@endpbox     % <fix> arydshln
2276     \let\adl@act@@endpbox    \tabu@adl@act@endpbox    % <fix> arydshln
2277     \let\@preamerror        \@preamerr               % <fix> arydshln
2278 }% \tabu@adl@fix

```

## longtable \@startpbox: \everypar needed

**\tabu@LT@startpbox** The leading strut should be inserted at \everypar in order for \tabulinesep to work (otherwise, T<sub>E</sub>X is in horizontal mode and \nointerlineskip breaks).

```

2279 \def\tabu@LT@startpbox #1{%
2280     \bgroup
2281     \let\@footnotetext\LT@p@ftntext
2282     \setlength\hsize{#1}%
2283     \@arrayparboxrestore
2284     \everypar{%
2285         \vrule \@height \ht\@arstrutbox \@width \z@
2286         \everypar{}}}%
2287 }% \tabu@LT@startpbox

```

## 11.23 Package options and Initialisation

### \tracingtabu and the package options

**delarray (package option)** The delarray package option is only there for convenience: it simply loads the delarray package.

```

2288 \DeclareOption{delarray}{\AtEndOfPackage{\RequirePackage{delarray}}}

```

**linegoal (package option)** The linegoal package option only sets \tabudefaultright to be equal to \linegoal. The required package linegoal is loaded.

```

2289 \DeclareOption{linegoal}{%
2290     \AtEndOfPackage{%

```

```

2291 \RequirePackage{linegoal}[2010/12/07]%
2292 \let\tabubefaulttarget \linegoal% \linegoal is \linewidth if not pdfTeX
2293 }}

```

**\scantokens (package option)** The `scantokens` package option makes `tabu` equal to `tabu*`.

```

2294 \DeclareOption{scantokens}{\tabuscantokenstrue}

```

**\tracingtabu** `\tracingtabu` is the same as the package option `debugshow`.

**debugshow (package option)**

```

2295 \DeclareOption{debugshow}{\AtEndOfPackage{\tracingtabu=\tw@}}
2296 \def\tracingtabu {\begingroup\@ifnextchar=%
2297   {\afterassignment\tabu@tracing\count@}
2298   {\afterassignment\tabu@tracing\count@1\relax}}
2299 \def\tabu@tracing{\expandafter\endgroup
2300   \expandafter\tabu@tr@cing \the\count@ \relax
2301 }% \tabu@tracing
2302 \def\tabu@tr@cing #1\relax {%
2303   \ifnum#1>\thr@@ \let\tabu@tracinglines\message
2304   \else \let\tabu@tracinglines\@gobble
2305   \fi
2306   \ifnum#1>\tw@ \let\tabu@DBG \tabu@@DBG
2307   \def\tabu@mkarstrut {\tabu@DBG@arstrut}%
2308   \tabustrutrul 1.5\p@
2309   \else \let\tabu@DBG \@gobble
2310   \def\tabu@mkarstrut {\tabu@arstrut}%
2311   \tabustrutrul \z@
2312   \fi
2313   \ifnum#1>\@ne \let\tabu@debug \message
2314   \else \let\tabu@debug \@gobble
2315   \fi
2316   \ifnum#1>\z@
2317     \let\tabu@message \message
2318     \let\tabu@tracing@save \tabu@message@save
2319     \let\tabu@starttimer \tabu@pdftimer
2320   \else
2321     \let\tabu@message \@gobble
2322     \let\tabu@tracing@save \@gobble
2323     \let\tabu@starttimer \relax
2324   \fi
2325 }% \tabu@tr@cing

```

## Initialisation and setup \AtBeginDocument

At the end of the `tabu` package:

- `\tracingtabu` is set to 0: this initialises the message commands. Eventually, the value will be overwritten by the `debugshow` package option later.
- `\everyrow` is set to empty: this initialises the process at `\everycr` to the default process,
- a new *empty* line style is defined, to be equivalent to `\hline`: this creates the *default leaders*, which will be used if a line style specification cannot be parsed successfully.

Then this default line style is set to be the current one.

At Begin Document, a fix for `arydshln` and `colortbl` compatibility with `delarray` shortcuts available inside `tabu`: requirement for this fix is checked by `\tabu@fix@arrayright`.

Then the switch `\iftabu@colortbl` is set.

Finally, the `longtabu` environment is defined only if the `longtable` package is detected.

```

2326 \AtBeginDocument{\tabu@AtBeginDocument}
2327 \def\tabu@AtBeginDocument{\let\tabu@AtBeginDocument \@undefined
2328   \ifdefined\arrayrulecolor \tabu@colortbltrue % <colortbl>

```

```

2329             \tabu@colortblalignments % different glues are used
2330 \else             \tabu@colortblfalse \fi
2331 \ifdefined\CT@arc@ \else \let\CT@arc@ \relax \fi
2332 \ifdefined\CT@drsc@\else \let\CT@drsc@ \relax \fi
2333 \let\tabu@arc@L \CT@arc@ \let\tabu@drsc@L \CT@drsc@
2334 \ifodd 1\ifcsname siunitx_table_collect_begin:Nn\endcsname % <siunitx: ok>
2335     \expandafter\ifx
2336         \csname siunitx_table_collect_begin:Nn\endcsname\relax 0\fi\fi\relax
2337     \tabu@siunitxtrue
2338 \else \let\tabu@maybesiunitx \@firstofone % <not siunitx: setup>
2339     \let\tabu@siunitx \tabu@nosunitx
2340     \tabu@siunitxfalse
2341 \fi
2342 \ifdefined\adl@array % <arydshln>
2343 \else \let\tabu@adl@fix \relax
2344     \let\tabu@adl@endtrial \@empty \fi
2345 \ifdefined\longtable % <longtable>
2346 \else \let\longtabu \tabu@nolongtabu \fi
2347 \ifdefined\cellspacetoplimit \tabu@warn@cellspace\fi
2348 \csname\ifcsname ifHy@hyperfootnotes\endcsname % <hyperfootnotes>
2349     ifHy@hyperfootnotes\else iffalse\fi\endcsname
2350 \let\tabu@footnotetext \tabu@Hy@ftntext
2351 \let\tabu@xfootnote \tabu@Hy@xfootnote \fi
2352 \ifdefined\FV@DefineCheckEnd% <fancyvrb>
2353     \tabu@fancyvrb \fi
2354 \ifdefined\color % <color / xcolor>
2355     \let\tabu@color \color
2356     \def\tabu@leavevmodecolor ##1{%
2357         \def\tabu@leavevmodecolor {\leavevmode ##1}%
2358     }\expandafter\tabu@leavevmodecolor\expandafter{\color}%
2359 \else
2360     \let\tabu@color \tabu@nocolor
2361     \let\tabu@leavevmodecolor \@firstofone \fi
2362 \tabu@latextwoe
2363 \ifdefined\@raggedtwoe@everyselectfont % <ragged2e>
2364     \tabu@raggedtwoe
2365 \else
2366     \let\tabu@cell@L \tabu@cell@l
2367     \let\tabu@cell@R \tabu@cell@r
2368     \let\tabu@cell@C \tabu@cell@c
2369     \let\tabu@cell@J \tabu@cell@j \fi
2370 \expandafter\in@ \expandafter\@arrayright \expandafter{\endarray}%
2371 \ifin@ \let\tabu@endarray \endarray
2372 \else \tabu@fix@arrayright \fi% <fix for colortbl & arydshln (delarray)>
2373 \everyrow{}%
2374 }% \tabu@AtBeginDocument
2375 \def\tabu@warn@cellspace{%
2376     \PackageWarning{tabu}{%
2377         Package cellspace has some limitations
2378     \MessageBreak And redefines some macros of array.sty.
2379     \MessageBreak Please use \string\tabulinesep\space to control
2380     \MessageBreak vertical spacing of lines inside tabu environnement}%
2381 }% \tabu@warn@cellspace

```

**\ProcessOption \*** is much quicker than without the star...

```

2382 \tabuscantokensfalse
2383 \let\tabu@arc@G \relax
2384 \let\tabu@drsc@G \relax

```

```

2385 \let\tabu@evr@G          \@empty
2386 \let\tabu@rc@G           \@empty
2387 \def\tabu@ls@G           {\tabu@linestyle@}%
2388 \let\tabu@@rowfontreset  \@empty % <init>
2389 \let\tabu@@celllalign    \@empty
2390 \let\tabu@@celllralign    \@empty
2391 \let\tabu@@cellleft       \@empty
2392 \let\tabu@@cellright      \@empty
2393 \def\tabu@naturalXmin     {\z@}
2394 \def\tabu@naturalXmax     {\z@}
2395 \let\tabu@rowfontreset    \@empty
2396 \def\tabulineon {4pt}\let\tabulineoff \tabulineon
2397 \tabu@everyrowtrue
2398 \ifdefined\pdfelapsedtime      % <pdfTeX>
2399     \def\tabu@pdftimer {\xdef\tabu@starttime{\the\pdfelapsedtime}}%
2400 \else     \let\tabu@pdftimer \relax \let\tabu@message@etime \relax
2401 \fi
2402 \tracingtabu=\z@
2403 \newtabulinestyle {=\maxdimen}% creates the 'factory' settings \tabu@linestyle@
2404 \tabulinestyle{}
2405 \taburowcolors{}
2406 \let\tabudefaultttarget \linewidth
2407 \ProcessOptions*              % \ProcessOptions* is quicker !
2408 </package>

```

## 12 References

- [1] *A new implementation of L<sup>A</sup>T<sub>E</sub>X's tabular and array environments* by Frank Mittelbach  
2008/09/09 v2.4c – Tabular extension package (FMI)  
[CTAN:help/Catalogue/entries/array.html](http://CTAN:help/Catalogue/entries/array.html)
- [2] *The varwidth package* by Donald Arseneau  
2009/03/30 ver 0.92 – Variable-width minipages  
[CTAN:help/Catalogue/entries/varwidth.html](http://CTAN:help/Catalogue/entries/varwidth.html)
- [3] *The enumitem-zref package* by  $\text{FC}$   
2010/11/28 ver 1.1 – Extended references for enumitem pkg  
[CTAN:help/Catalogue/entries/enumitem-zref.html](http://CTAN:help/Catalogue/entries/enumitem-zref.html)

## 13 History

[2011/02/19 v2.5]

- Bug fixed for `\pdfelapsedtime` when compilation without pdfL<sup>A</sup>T<sub>E</sub>X.
- Modification of `\@finalstrut` (“null-rule” added) to avoid problems with `\columncolor`.

[2011/02/17 v2.4]

- Documentation revisited

[2011/02/13 v2.3]

- Fixed two bugs for nested `tabu` environment: when using `\rowfont` and when `tabu` is nested inside `longtabu`

[2011/02/12 v2.2 – New implementation - Absolutely no modification of `array.sty`]

- $\tau_{\text{nb}} \subset$  has been totally reimplemented, including the algorithms.  
In particular, outside of the `tabu` environment, absolutely none of the macros of `array.sty`, (and obviously none of L<sup>A</sup>T<sub>E</sub>X) is modified.

The process has been completely reinvented: `tabu` follows a path along different modes (or strategies) measuring natural width of cells, fixing **X** column widths, measuring vertical length of rows and then printing the final tabular. The process is optimized, especially in the case of nested `tabu` environments: a tabular is not built twice for measuring purpose... As a result, many new features are now possible... vertical leaders (dashed lines), dynamic vertical spacing adjustment, and hopefully still more in a next release.

`tabu` now systematically collects the environment body. But with `\scantokens`, it is possible to insert verbatim material inside the columns: use `tabu*` instead of `tabu`, for the outer most tabular.

- New: `\firsthline` and `\lasthline` can draw multiple lines, and there is an option to set `\extratabsurround` instantly, and locally.
- New: `\taburulecolor` with a good behaviour with groupings (like `\everyrow`)
- Modification: `\tabulinestyle` sets the line style for the `tabu`, `\newtabulinestyle` defines a new line style.

This	is	the	new	$\tau_{\text{nb}} \subset$	package
------	----	-----	-----	----------------------------	---------

### [2011/01/19 v2.1]

- [Vertical spacing](#) had a bug with `longtabu` and paragraph columns. Fixed.
- New: `\everyrow`.
- Fix a bug of `\rowfont` when using `siunitx` S columns.
- Some code optimisation.
- To do (if possible): a syntax `X[6mc]S[...]` to “embed” `siunitx` S column inside `tabu` and `longtabu` X columns...

### [2011/01/18 v2.0]

- [Vertical spacing](#) of lines implemented ! See `\tabulinesep` and `\extrarowsep`.
- `\tabulinestyle` : user defined line style can now be used inside the optional argument of the `[[...]` preamble token.
- `[[...]` is now allowed in `\multicolumn` preamble inside `tabu` environment.
- Bug fixed inside `\tabu@prepnexttok` (again !!! - a difficult case !)
- Incompatibility of package `cellspace` with `tabu spread` and `tabu` with [negativ coefficients](#) for X columns with has been lifted.  
However, as said in the documentation of package `cellspace`, S column modifier does not work in the case of nested tabulars.  
The S column modifier becomes C when the package `siunitx` is loaded (see `siunitx` documentation).  
Moreover, `cellspace` does not work with `color` or `xcolor` and paragraph column types !!  
Finally, `cellspace` redefines globally `\@startpbox` and `\@endpbox` and is therefore not fully compatible with `array.sty` and therefore with  $\mathcal{T}_\mathbb{N}b\subset$ .  
For all those reasons,  $\mathcal{T}_\mathbb{N}b\subset$  displays a warning to discourage the use of `cellspace` with the `tabu` environment.

### [2011/01/15 v1.9]

- Bug in `\savetabu` when used inside `longtabu`...
- Bug when `tabu` with X column is nested inside `lontabu`.
- Documentation (`\rowfont` was missing in the [summary](#)).

### [2010/12/28 v1.8]

- `\tracingtabu` / `debugshow` package option:  
reporting of the time elapsed during trials (if `\pdfelapsedtime` and thus pdfT<sub>E</sub>X is available)  
Slight modifications for better reporting on the .log file.
- Fix a bug when `\savetabu` is used after `\multicolumn` (`\multicolumn` globally redefines `\@preamble`).
- Fix a bug with `\tabucline` and `\CT@arc@` (`colortbl`).
- Better privacy of columns types specifically defined for `tabu`.
- Improvement in the rewriting process (but only very few people should notice...)
- Documentation.

### [2010/12/18 v1.7]

- Code optimisation
- Modification in the columns rewriting process (bug with some new column types defined by the user).

### [2010/12/07 v1.5]

- Implementation of negativ width coefficients for **X** columns (cf. [tabu X columns – Mastering horizontal space point 2](#)).
- Columns natural widths computation (for **tabu spread** with **X** columns and negativ coefficients) is based on the code of the **varwidth** package by Donald Arseneau.
- **longtabu** is now provided, based on the **longtable** package by David Carlisle. **longtabu** can be used just like **tabu**.
- Vertical lines can be used whatever the catcode of **|** is.
- **\savetabu** reports saved informations in the .log ([debugshow](#) option).
- **\savetabu... \usetabu** now restores the **\halign** preamble rather than the **tabu** preamble! **\preamble** can be use in the **tabu** preamble to restore a **tabu** preamble.
- **\tabucline** is more robust with “special” preambles containing **>** or **<** tokens. **\tabucline** now takes care of **\arrayrulecolor** (package **colortbl**).
- **enumitem-zref** package has been added to the documentation (see the link [point 1](#))
- Optimisation of some parts of the code.

### [2010/11/22 v1.4]

- Compatibility improvement with **linegoal** for the syntax: **\begin {tabu}** to **\linegoal {...}**
- Hyper footnotes now work correctly.
- Fix a bug when using colored vertical lines in **tabu** in math mode.
- Fix a bug with vertical lines and **colortbl \arrayrulecolor** specification.
- Fix a compatibility bug with **arydshln**: when nesting a tabular that use vertical dashed lines (**arydshln**) inside **tabu spread** with **X** columns.

### [2010/11/18 v1.3]

- Fix a bug that may appear in **\tabucline** depending on the preamble due to arbitrary **\countdef**.
- Improvement in the use of **\everycr**: no **\global** stuff. Thus bug fixed when nesting **tabu** inside **AMS-align** environment for example. Same issue with **\rowfont** which now works without global modification of **\everycr**.
- No phantom line is added to **tabu** but a command **\tabuphantomline** is provided for this purpose (required with **\multicolumn** in some cases).
- Improvement on vertical alignment.
- To do: an example file to test a wide range of possibilities...
- Documentation.



## [2010/11/15 v1.2]

- Improvement in parameters parsing for optional parameters (`|` and `\tabucline`).
- Modification / optimization in `\tabu@prepnext@tok`.
- Modification of `\tabucline` to get better results with `m` columns (`X[m]`) and also when `\minrowclearance > 0` (package `colortbl`).

## [2010/10/28 v1.1]

- First version.

# 14 Index

Numbers written in *italic* refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; plain numbers refer to the code lines where the entry is used.

Symbols	
<code>\&amp;</code> . . . . .	80
<code>\@arrayparboxrestore</code> . . . . .	2283
<code>\@arrayright</code> . . . . .	1027, 1034, 1041, 2226, 2231, 2236, 2243, 2245, 2247, 2370
<code>\@arraystretch</code> . . . . .	688
<code>\@arstrutbox</code> 109, 117, 118, 123, 620, 686, 687, 814, 815, 996, 1272, 1541, 1624, 1640, 1712, 1716, 1719, 1725, 1728, 1730, 2269, 2285	
<code>\@defaultunits</code> 329, 330, 851, 1056, 1755, 1799, 1800	
<code>\@endpbox</code> . . . . .	611, 633
<code>\@finalstrut</code> . . . . .	1029, 1262, 1541, 1617, 2269
<code>\@flushglue</code> . . . . .	1066, 1530, 2088
<code>\@footnotemark</code> . . . . .	2166
<code>\@footnotetext</code> . . . . .	622, 1120, 2281
<code>\@gobbletwo</code> . . . . .	840, 1387, 1511, 1794
<code>\@halignto</code> . . . . .	866, 875, 1034, 1124, 1255, 1268, 1269, 1954
<code>\@ifnextchar</code> . . . . .	166, 168, 177, 197, 199, 208, 237, 285, 847, 2296
<code>\@let@token</code> . . . . .	62, 63, 64, 156, 158, 170, 179, 201, 210, 239, 558, 560, 561, 562, 563, 888, 891, 892, 893, 894, 895, 896, 897, 898, 899, 900, 901, 902, 903, 904, 905, 906, 907, 945, 947
<code>\@mkpream</code> . . . . .	827
<code>\@normalcr</code> . . . . .	621, 2207
<code>\@onelevel@sanitize</code> . . . . .	97
<code>\@preamble</code> . . . . .	1070, 1074, 1078, 1085, 1490, 1925, 1932, 2232, 2237, 2244
<code>\@raggedtwoe@everyselectfont</code> . . . . .	2363
<code>\@rightskip</code> . . . . .	2087
<code>\@sharp</code> . . . . .	1939
<code>\@sptoken</code> . . . . .	64, 561, 904, 1058, 1818, 1852
<code>\@tabarray</code> . . . . .	612, 634
<code>\@tempb</code> . . . . .	1894
<code>\@tempcnta</code> . . . . .	25, 1116, 1737, 2010, 2015
<code>\@tempcntb</code> . . . . .	26, 1738
<code>\@tempdimb</code> . . . . .	36, 116, 118, 1331, 1337, 1348, 1351, 1358, 1388, 1391, 1451, 1455, 1460, 1463, 1601, 1604, 1605, 1613, 1616, 1618, 1706, 1718, 1725, 1728, 1730
<code>\@tempdimc</code> 34, 37, 1373, 1375, 1379, 1381, 1384, 1427, 1428, 1611, 1705, 1725, 1731	
<code>\@temptokena</code> 739, 744, 760, 850, 854, 1010, 1013, 1057, 1810, 1826, 1830, 1861, 1864	
<code>\@thefnmark</code> . . . . .	2164, 2173
<code>\@trivlist</code> . . . . .	2208, 2209
<code>\@undefined</code> 6, 403, 502, 608, 1771, 1953, 1961, 2327	
<code>\@whiles</code> . . . . .	1345
<code>\@xfootnote</code> . . . . .	623, 1120, 2159
<code>\@xfootnotetext</code> . . . . .	622
<code>\@xhline</code> . . . . .	619
<code>\{</code> . . . . .	1891
<code>\}</code> . . . . .	1891
A	
<code>\abovetabulinesep</code> . . . . .	39, 200, 202, 204, 209, 211, 213, 216, 217, 1600, 1612
<code>\active</code> . . . . .	445, 446, 447
<code>\adl@endarray</code> . . . . .	2230, 2242
<code>\afterassignment</code> . . . . .	142, 173, 176, 204, 207, 238, 240, 241, 482, 573, 940, 1231, 2210, 2217, 2297, 2298
<code>\aftergroup</code> . . . . .	146, 163, 164, 194, 195, 652, 655, 827, 989, 990, 1231, 1532, 1791, 2142, 2212, 2220
<code>\array</code> . . . . .	539
<code>\arraybackslash</code> . . . . .	1530, 2196, 2197, 2198, 2202, 2203, 2204, 2205
<code>\arraycolsep</code> . . . . .	729
<code>\arrayrulecolor</code> . . . . .	2328
<code>\arrayrulewidth</code> . . . . .	. . . 428, 731, 1498, 1668, 1692, 1693, 2259
<code>\arraystretch</code> . . . . .	110, 111, 115, 116, 126, 131, 736, 1503, 1635
B	
<code>\baselineskip</code> . . . . .	227, 1981
<code>\begin</code> . . . . .	8, 9
<code>\belowtabulinesep</code> . . . . .	39, 200, 202, 209, 211, 213, 216, 217, 1601, 1613
C	
<code>\c@footnote</code> . . . . .	2160
<code>\c@page</code> . . . . .	1410, 1412
<code>\c@taburow</code> . . . . .	21, 254, 348, 350, 351, 368, 372, 680, 716, 751, 1622, 1623, 1625, 1627, 1628, 1629, 1630, 1632, 1636
<code>\c@taburow@save</code> . . . . .	680, 716
<code>\catcode</code> 10, 11, 80, 92, 445, 446, 447, 448, 1890	
<code>\Centering</code> . . . . .	625, 917, 2099, 2101, 2202

`\centering` . . . 623, 913, 917, 2070, 2072, 2196  
`\cl@ckpt` . . . . . 1132  
`\cleaders` . . . . . 1763  
`\cline` . . . . . 1688, 1693, 1698  
`\color` . . . . . 90, 122, 299, 303, 312, 319,  
 506, 620, 1121, 1489, 1997, 2354, 2355, 2358  
`\color@` . . . . . 487  
`\color@b@x` . . . . . 628  
`\color@begingroup` . . . . . 1539  
`\color@block` . . . . . 2216  
`\colorlet` . . . . . 353, 365, 370  
`\CT@do@color` . . . . . 700  
`\CT@everycr` . . . . . 636  
`\currentgrouplevel` . . . . . 225, 228, 229, 231  
`\currentgrouptype` . . . . .  
 . . . . . 1586, 1591, 1592, 1984, 2117, 2122

## D

`\d@llarbegin` . . . . . 1589  
`\d@llarend` . . . . . 136, 1599  
`\debugshow␣(package␣option)` . . . . . 2295  
`\definecolorseries` . . . . . 349, 363  
`\delarray␣(package␣option)` . . . . . 2288  
`\detokenize` . . . . . 1489,  
 1507, 1508, 1509, 1884, 1887, 1888, 1893  
`\dimen@` . . . . . 1135, 1136,  
 1137, 1142, 1303, 1306, 1310, 1315, 1353,  
 1355, 1361, 1371, 1372, 1397, 1417, 1681,  
 1707, 1712, 1725, 1728, 1730, 2222, 2223  
`\displaystyle` . . . . . 947  
`\doublerulesep` . . . 732, 1499, 1677, 1707, 1785  
`\dp` . . . . . 111, 116, 118, 131,  
 687, 814, 815, 1601, 1613, 1624, 1634,  
 1716, 1719, 1725, 1728, 1730, 2216, 2223

## E

`\ecolumn` . . . . . 1538  
`\endarray` . . . . . 539, 635, 1154,  
 1213, 1242, 1277, 2226, 2248, 2370, 2371  
`\endlinechar` . . . . . 100, 401, 1127  
`\endlongtable` . . . . . 549  
`\endlongtabu` . . . . . 549  
`\endtabu` . . . . . 538, 1579  
`\endtabular` . . . . . 540  
`\errmessage` . . . . . 1301  
`\everycr` . . . . . 610, 629, 1228, 1229, 1232  
`\everypar` 552, 629, 641, 642, 1261, 1585, 2284, 2286  
`\everyrow` . . . . . 17, 234, 385, 724, 757, 2373  
`\extracolsep` . . . . . 1066  
`\extrarowdepth` 39, 111, 116, 131, 169, 171, 173,  
 178, 180, 182, 185, 186, 687, 735, 1502, 1634  
`\extrarowheight` . . . 110, 115, 126, 169, 171,  
 178, 180, 182, 185, 186, 686, 734, 1501, 1633  
`\extrarowsep` . . . . . 13, 162, 384  
`\extratabsurround` . . . . .  
 . . . 384, 733, 1500, 1685, 1686, 1715, 1731

## F

`\FancyVerbFormatLine` . . . . . 1901  
`\firstline` . . . . . 15, 617, 1687, 1713, 1722  
`\firstline` . . . . . 618  
`\footnotetext` . . . . . 2154, 2158  
`\futurelet` . . . . . 62, 156, 558,  
 574, 888, 945, 1669, 1689, 1709, 1751, 1814  
`\FV@@@CheckEnd` . . . . . 1879

`\FV@@CheckEnd` . . . . . 1878, 1887  
`\FV@CheckEnd` . . . . . 1877  
`\FV@DefineCheckEnd` . . . . . 1869, 1885, 2352  
`\FV@endScanning` . . . . . 1880  
`\FV@EnvironName` . . . . . 1881, 1882, 1884  
`\FV@LeftListFrame` . . . . . 1900  
`\FV@LeftListNumber` . . . . . 1899  
`\FV@ListProcessLine` . . . . . 1529

## G

`\GenericError` . . . . . 150, 1118  
`\globaldefs` . . . . . 691, 718, 1132, 1639, 1654  
`\gluestretchorder` . . . . . 2132, 2137

## H

`\H@@footnotetext` . . . . . 2181  
`\hbadness` . . . . . 1116  
`\hfilneg` . . . . . 2138  
`\hfuzz` . . . . . 616, 1117  
`\hline` . . . . . 619, 1663, 1665,  
 1666, 1672, 1687, 1688, 1692, 1697, 1711  
`\hrule` . . . . . 438, 515, 1618, 1668, 1675, 1783  
`\ht` 110, 115, 117, 126, 516, 686, 814, 815, 1600,  
 1612, 1624, 1633, 1712, 2216, 2222, 2285

## I

`\ifcase` . . . . . 450, 458, 471, 560, 602, 717,  
 719, 891, 1021, 1043, 1094, 1172, 1258,  
 1280, 1290, 1706, 1817, 1834, 2015, 2029  
`\ifcat` . . . . . 136, 229, 404, 459, 472, 804, 907  
`\ifcsname` . . . 94, 388, 393, 487, 1993, 2334, 2348  
`\ifdim` . . . . . 117, 118,  
 122, 227, 420, 422, 424, 428, 582, 592,  
 603, 639, 644, 869, 954, 955, 966, 967,  
 968, 1048, 1136, 1138, 1142, 1145, 1271,  
 1306, 1309, 1314, 1321, 1334, 1335, 1354,  
 1357, 1379, 1381, 1397, 1404, 1418, 1427,  
 1434, 1450, 1451, 1460, 1521, 1525, 1543,  
 1544, 1551, 1553, 1602, 1603, 1604, 1614,  
 1615, 1616, 1952, 1973, 1981, 2118, 2136  
`\iffalse` . . . . . 56, 594, 1126, 1572  
`\ifin@` . . . . . 433, 2371  
`\ifnum` . . . . . 79, 103, 106, 331, 333,  
 348, 350, 351, 368, 594, 1092, 1093, 1124,  
 1126, 1168, 1271, 1329, 1402, 1410, 1448,  
 1473, 1474, 1475, 1477, 1478, 1572, 1586,  
 1591, 1592, 1622, 1667, 1679, 1681, 1721,  
 1734, 1739, 1741, 1744, 1789, 1791, 1793,  
 1801, 1802, 1803, 1807, 1957, 1964, 1982,  
 1984, 1995, 2009, 2117, 2122, 2132, 2137,  
 2253, 2254, 2255, 2303, 2306, 2313, 2316  
`\ifodd` . . . . . 82, 117, 268, 404,  
 424, 592, 650, 852, 1519, 1576, 1949, 2334  
`\iftabu@cellright` . . . . . 2007, 2017, 2025  
`\iftabu@colortbl` 49, 249, 636, 2116, 2130, 2227  
`\iftabu@everyrow` . . . . . 49, 103, 105, 163,  
 194, 255, 273, 297, 310, 338, 346, 636, 1233  
`\iftabu@long` . . . . . 49, 650, 989, 1205  
`\iftabu@measuring` . . . . . 49,  
 581, 864, 1030, 1036, 1093, 1291, 1345, 1374  
`\iftabu@negcoef` . . . . . 49, 1195, 1920  
`\iftabu@siunitx` . . . . . 49, 999  
`\iftabu@spread` . . . 49, 592, 867, 1190, 1286, 1520  
`\iftabuscantokens` . . . . .  
 . . . 58, 842, 1127, 1154, 1209, 1218, 1235

<code>\iftrue</code> . . . . .	55	<code>\PackageWarning</code> . . . . .	
<code>\ignorespaces</code> . . . . .	1817, 1851	. . . . .	151, 495, 928, 935, 1970, 2246, 2376
<code>\immediate</code> . . . . .	144, 146	<code>\parfillskip</code> . . . . .	1530, 2088
<code>\in@</code> . . . . .	334, 336, 337, 2033, 2035, 2370	<code>\pdfadjustspacing</code> . . . . .	1531
<code>\in@false</code> . . . . .	426	<code>\pdfelapsedtime</code> . . . . .	1442, 2398, 2399
<code>\in@true</code> . . . . .	425	<code>\preamble</code> . . . . .	20, 981, 987, 1005, 1959
<code>\indent</code> . . . . .	600	<code>\preamble_(private_column_type)</code> . . . . .	981
<b>J</b>			
<code>\justifying</code> . . . . .	627, 919, 2111, 2113, 2205	<code>\prepnext@tok</code> . . . . .	630, 2039
<b>L</b>			
<code>\lastbox</code> . . . . .	643	<code>\prevdepth</code> . . . . .	1611
<code>\lasthline</code> . . . . .	617, 1688, 1726, 1735	<code>\protected@xdef</code> . . . . .	2164, 2173
<code>\lastline</code> . . . . .	618	<b>R</b>	
<code>\lastskip</code> . . . . .	2131, 2136, 2137	<code>\RaggedLeft</code> . . . . .	626, 918, 2105, 2107, 2203
<code>\lccode</code> . . . . .	445	<code>\raggedleft</code> . . . . .	624, 914, 918, 2076, 2078, 2197
<code>\leaders</code> . . . . .	438, 1762	<code>\RaggedRight</code> . . . . .	626, 916, 2093, 2095, 2204
<code>\leftmargin</code> . . . . .	1897	<code>\raggedright</code> . . . . .	624, 912, 916, 2064, 2066, 2198
<code>\leftskip</code> . . . . .	2087	<code>\rem@pt</code> . . . . .	1431, 1435, 1438
<code>\linegoal</code> . . . . .	1406, 2292	<code>\remove@to@nnil</code> . . . . .	90
<code>\linegoal_(package_option)</code> . . . . .	2289	<code>\resetcolorseries</code> . . . . .	352, 364, 369
<code>\linewidth</code> . . . . .	606, 639, 1119, 1403, 1404, 1405, 1898, 2292, 2406	<code>\rightskip</code> . . . . .	2087
<code>\longtable</code> . . . . .	549, 2345	<code>\romannumeral</code> . . . . .	1470, 1487, 1756
<code>\longtabu</code> . . . . .	76, 547, 555, 2346	<code>\rowcolor</code> . . . . .	354, 366, 371, 1121
<code>\lowercase</code> . . . . .	446, 1468	<code>\rowfont</code> . . . . .	18, 627, 1980
<code>\LT@bchunk</code> . . . . .	1082, 1087	<b>S</b>	
<code>\LT@echunk</code> . . . . .	1221	<code>\save@decl</code> . . . . .	137
<code>\LT@get@widths</code> . . . . .	1223	<code>\savetabu</code> . . . . .	19, 739, 1119, 1513, 1908, 1975
<code>\LT@p@ftntext</code> . . . . .	2281	<code>\scantokens</code> . . . . .	59, 100, 411, 415, 1079, 1088
<code>\LT@startpbox</code> . . . . .	552	<code>\scantokens_(package_option)</code> . . . . .	2294
<code>\LTchunksize</code> . . . . .	550	<code>\setbox</code> . . . . .	109, 123, 526, 530, 643, 813, 996, 1206, 1215, 1222, 1272, 1527, 1587, 1594, 1640, 2210, 2217
<b>M</b>			
<code>\maxdimen</code> . . . . .	402, 406, 420, 422, 424, 428, 967, 1117, 1340, 2403	<code>\setlength</code> . . . . .	870, 2282
<code>\message</code> . . . . .	2303, 2313, 2317	<code>\span</code> . . . . .	1745, 1746
<code>\minrowclearance</code> . . . . .	737, 1504	<code>\stretch</code> . . . . .	2117, 2119
<code>\multicolumn</code> . . . . .	72, 631, 1188	<code>\strutbox</code> . . . . .	110, 111, 115, 116, 126, 131, 686, 687, 1633, 1634
<code>\multispan</code> . . . . .	837, 840, 1188	<b>T</b>	
<b>N</b>			
<code>\narrowragged</code> . . . . .	1530	<code>\tabcolsep</code> . . . . .	730
<code>\NC@find</code> . . . . .	588, 790, 833, 846, 853, 854, 858, 1010, 1013, 1022, 1026, 1061, 1509	<code>\tabu</code> . . . . .	75, 538, 1569
<code>\NC@rewrite@S</code> . . . . .	855	<code>\tabu@</code> 409, 411, 416, 479, 480, 481, 484, 493, 539, 540, 549, 589, 958, 992, 995, 1037, 1071, 1083, 1135, 1303, 1336, 1371, 1921	
<code>\newcount</code> . . . . .	21, 22, 23, 24, 27, 28	<code>\tabu@@</code> . . . . .	1071, 1083, 1113
<code>\newdimen</code> . . . . .	30, 31, 32, 33, 38, 39, 40, 41, 42	<code>\tabu@@@celllalign</code> . . . . .	
<code>\newsavebox</code> . . . . .	45, 46, 47, 48	. . . . .	672, 706, 1641, 1655, 1987, 2003, 2389
<code>\newskip</code> . . . . .	1980	<code>\tabu@@@celllalign@save</code> . . . . .	672, 706
<code>\newtabulinestyle</code> . . . . .	16, 263, 2403	<code>\tabu@@@cellleft</code> . . . . .	
<code>\newtoks</code> . . . . .	43, 44	. . . . .	674, 708, 1643, 1657, 1985, 2000, 2391
<code>\noalign</code> . . . . .	103, 189, 220, 227, 247, 362, 366, 1226, 1233, 1638, 1653, 1667, 1681, 1713, 1718, 1726, 1728, 1739, 1781, 1908, 1981	<code>\tabu@@@cellleft@save</code> . . . . .	674, 708
<code>\noindent</code> . . . . .	1605, 1651, 1652	<code>\tabu@@@cellralign</code> . . . . .	
<code>\numexpr</code> 79, 162, 193, 1444, 1445, 1446, 1793, 2015		. . . . .	673, 707, 1642, 1656, 1988, 2004, 2390
<b>O</b>			
<code>\obeyspaces</code> . . . . .	69, 410, 1127	<code>\tabu@@@cellralign@save</code> . . . . .	673, 707
<code>\omit</code> . . . . .	1714, 1729, 1744, 1745, 1746	<code>\tabu@@@cellright</code> . . . . .	
<b>P</b>			
<code>\PackageError</code> . . . . .	379, 548, 556, 860, 1977	. . . . .	675, 709, 1644, 1658, 1986, 2001, 2392
<code>\PackageInfo</code> . . . . .	1974	<code>\tabu@@@cellright@save</code> . . . . .	675, 709
		<code>\tabu@@@DBG</code> . . . . .	122, 2306
		<code>\tabu@@everycr</code> . . . . .	629, 1232
		<code>\tabu@@everypar</code> . . . . .	629
		<code>\tabu@@@rowfontreset</code> . . . . .	
		. . . . .	677, 711, 1989, 1999, 2005, 2388
		<code>\tabu@@@rowfontreset@save</code> . . . . .	677, 711
		<code>\tabu@@@spxiii</code> . . . . .	71, 1160
		<code>\tabu@@@tabudecimal</code> . . . . .	1832

<code>\tabu@@thevline</code>	1645, 1659	<code>\tabu@cellrighttrue</code>	2023
<code>\tabu@adl@fix</code>	635, 2273, 2278, 2343	<code>\tabu@cellskip</code>	1980, 2118, 2131, 2132, 2133
<code>\tabu@afterendpar</code>	603, 721, 726, 1044	<code>\tabu@cellspacepatch</code>	1533, 1538
<code>\tabu@aftergroupcleanup</code>	651, 652, 654, 656, 1071, 1076, 1083, 1087, 1122	<code>\tabu@Centering</code>	625, 2202
<code>\tabu@align</code>	588, 589, 1928, 1936, 1956, 1958, 1963, 1965	<code>\tabu@centering</code>	623, 2196
<code>\tabu@aligndefault</code>	585, 604, 606, 1496, 1928, 1936, 1939, 1958, 1965, 1969	<code>\tabu@clarstrut</code>	685, 723
<code>\tabu@aligndefault@text</code>	1956, 1963, 1969	<code>\tabu@clckpt</code>	1132, 1208, 1217, 1268
<code>\tabu@alloc</code>	21, 679, 717, 756	<code>\tabu@cleanup</code>	649
<code>\tabu@alloc@</code>	29, 601	<code>\tabu@cline</code>	1739, 1740, 1752
<code>\tabu@alloc@save</code>	679, 756	<code>\tabu@clinearg</code>	1742, 1753, 1768
<code>\tabu@arc@G</code>	321, 658, 659, 692, 693, 745, 2383	<code>\tabu@clinebox</code>	1757, 1758, 1759, 1760, 1761, 1769
<code>\tabu@arc@Gsave</code>	658, 692	<code>\tabu@clineleads</code>	1762, 1763, 1764, 1769, 1770
<code>\tabu@arc@L</code>	315, 659, 740, 745, 752, 1505, 2333	<code>\tabu@clinescan</code>	1688, 1693
<code>\tabu@arith</code>	1287, 1320, 1393	<code>\tabu@closetrialsgroup</code>	1234
<code>\tabu@arith@negcoef</code>	1303, 1305, 1319	<code>\tabu@cnt</code>	21, 940, 942, 994, 1016, 1021, 1115, 1253, 1259, 1328, 1329, 1346, 1366, 1447, 1448, 1477
<code>\tabu@arithnegcoef</code>	1302, 1330, 1347, 1390	<code>\tabu@collectbody</code>	596, 1128, 1149, 1573
<code>\tabu@arstrut</code>	109, 2310	<code>\tabu@color</code>	506, 620, 1997, 2355, 2360
<code>\tabu@arstrutbox</code>	45, 620	<code>\tabu@color@b@x</code>	628, 2215, 2218
<code>\tabu@AtBeginDocument</code>	2326, 2327, 2374	<code>\tabu@colortblalignments</code>	2146, 2151, 2329
<code>\tabu@begin</code>	558	<code>\tabu@colortblfalse</code>	2330
<code>\tabu@begincolor@b@x</code>	2217, 2219	<code>\tabu@colortbltrue</code>	2328
<code>\tabu@beginfbox</code>	2210, 2211	<code>\tabu@commaXIII</code>	410, 447
<code>\tabu@box</code>	45, 1206, 1213, 1215, 1223, 1331, 1332, 1373, 1391, 1415, 1424, 1527, 1542, 1544, 1545, 1549, 1550, 1556, 1587, 1594, 1599, 1600, 1601, 1612, 1613, 1619	<code>\tabu@DBG</code>	125, 129, 1714, 1729, 2306, 2309
<code>\tabu@bufferX</code>	1250, 1256, 1257, 1284, 1294, 1298, 1366, 1376	<code>\tabu@DBG@arstrut</code>	122, 2307
<code>\tabu@C@extra</code>	162, 182, 186, 190, 191	<code>\tabu@debug</code>	1546, 1602, 1614, 2313, 2314
<code>\tabu@C@l@r</code>	83, 87, 90, 488, 489	<code>\tabu@decimal</code>	1810, 1864
<code>\tabu@C@linesep</code>	193, 213, 217, 221, 222	<code>\tabu@definestyle</code>	398, 401
<code>\tabu@C@loff</code>	403, 419, 443, 488	<code>\tabu@DELTA</code>	30, 1320, 1322, 1325, 1332, 1334, 1335, 1337, 1343, 1348, 1354, 1355, 1357, 1361, 1418, 1419
<code>\tabu@C@lon</code>	403, 417, 418, 443, 489	<code>\tabu@docline</code>	250, 1787, 1789
<code>\tabu@cell@align</code>	2054, 2063, 2069, 2075, 2081, 2092, 2098, 2104, 2110	<code>\tabu@docline@evr</code>	250, 1790
<code>\tabu@cell@C</code>	2097, 2102, 2368	<code>\tabu@doclineafter</code>	1790, 1791
<code>\tabu@cell@c</code>	2062, 2368	<code>\tabu@docolor@b@x</code>	2216, 2224
<code>\tabu@cell@J</code>	2109, 2114, 2369	<code>\tabu@dp</code>	1451, 1454, 1460, 1462, 1604, 1616, 1630, 1634
<code>\tabu@cell@j</code>	2062, 2369	<code>\tabu@dpdef</code>	1604, 1616, 1624, 1629
<code>\tabu@cell@L</code>	2091, 2096, 2366	<code>\tabu@drsc@G</code>	322, 660, 661, 694, 695, 746, 2384
<code>\tabu@cell@l</code>	2062, 2366	<code>\tabu@drsc@Gsave</code>	660, 694
<code>\tabu@cell@R</code>	2103, 2108, 2367	<code>\tabu@drsc@L</code>	316, 661, 741, 746, 753, 1506, 2333
<code>\tabu@cell@r</code>	2062, 2367	<code>\tabu@elapsedtime</code>	721, 1075, 1086, 1095, 1187
<code>\tabu@cellfont</code>	2002	<code>\tabu@end@envir</code>	1153, 1576, 1577, 1580
<code>\tabu@celllalign</code>	161, 668, 702, 1641, 1646, 1655, 1987, 2003, 2047, 2050, 2055	<code>\tabu@endarray</code>	635, 2229, 2234, 2241, 2371
<code>\tabu@celllalign@def</code>	161, 1099, 1104, 1108, 1182, 1261, 1265, 2052, 2056	<code>\tabu@endcolor@b@x</code>	2220, 2221, 2225
<code>\tabu@celllalign@save</code>	668, 702	<code>\tabu@endenvir</code>	1154, 1156, 1246, 1249, 1254, 1277
<code>\tabu@celleft</code>	158, 670, 704, 1643, 1649, 1657, 1850, 1985, 1992, 2000, 2048, 2050, 2059, 2060, 2064, 2070, 2076, 2082, 2093, 2099, 2105, 2111	<code>\tabu@endfbox</code>	2212, 2213
<code>\tabu@celleft@save</code>	670, 704	<code>\tabu@endofcollect</code>	1149
<code>\tabu@cellralign</code>	669, 703, 1100, 1104, 1108, 1183, 1261, 1266, 1642, 1647, 1656, 1988, 2004, 2044, 2050, 2057, 2058	<code>\tabu@endoftrials</code>	1173, 1177, 1234
<code>\tabu@cellralign@save</code>	669, 703	<code>\tabu@endpbox</code>	611, 633
<code>\tabu@cellright</code>	158, 671, 705, 1644, 1648, 1658, 1986, 2001, 2044, 2050	<code>\tabu@endpboxmeasure</code>	1532, 1540
<code>\tabu@cellright@save</code>	671, 705	<code>\tabu@endquick</code>	1573, 1575
<code>\tabu@cellrightfalse</code>	2013, 2018, 2026	<code>\tabu@endrewrite</code>	1021, 1025
		<code>\tabu@endrewritemulticolumn</code>	827, 835
		<code>\tabu@endtabu</code>	1576, 1583
		<code>\tabu@endtabus</code>	1577, 1584
		<code>\tabu@endtrial</code>	1203, 1279
		<code>\tabu@err</code>	1280, 1281, 1282, 1301
		<code>\tabu@everycr</code>	1228, 1233
		<code>\tabu@everyr@w</code>	241, 244, 260
		<code>\tabu@everyrow</code>	245, 1233
		<code>\tabu@everyrow@bgroup</code>	103, 234, 272, 283, 326
		<code>\tabu@everyrow@egroup</code>	103, 259, 281, 324, 376
		<code>\tabu@everyrowfalse</code>	56, 246, 1652



<code>\tabu@everyrowtrue</code>	55, 691, 751, 838, 2038, 2397	<code>\tabu@hsize</code>	961, <u>965</u> , 992, 1324, 1518, 1519, 1524
<code>\tabu@evr</code>	261, 262, 1099, 1104, 1108, 1182, 1193, 1260, 1264	<code>\tabu@hskip</code>	2128, 2150
<code>\tabu@evr@G</code>	257, 666, 667, 701, 724, 749, 2385	<code>\tabu@ht</code>	1450, 1452, 1457, 1603, 1615, 1628, 1633
<code>\tabu@evr@Gsave</code>	666, 701	<code>\tabu@htdef</code>	1603, 1615, 1624, 1627
<code>\tabu@evr@L</code>	256, 667, 742, 749, 757, 1507	<code>\tabu@Hy@footnotetext</code>	2175, 2180, 2193
<code>\tabu@evrh@C@k</code>	252, 261	<code>\tabu@Hy@ftntext</code>	<u>2160</u> , 2350
<code>\tabu@evrstartstop</code>	235, 237, 238, 240, 243	<code>\tabu@Hy@ftntxt</code>	2160, 2166, 2168, 2179
<code>\tabu@extr@</code>	176, 177, 184	<code>\tabu@Hy@xfootnote</code>	<u>2160</u> , 2351
<code>\tabu@extra</code>	166, 168, 175	<code>\tabu@if@envir</code>	73, 75
<code>\tabu@extrac@lsep</code>	1069	<code>\tabu@ifcolorleavevmode</code>	1997
<code>\tabu@extracolsep</code>	1049, <u>1056</u>	<code>\tabu@ifenvir</code>	<u>60</u> , 836
<code>\tabu@fancyvrb</code>	1873, 1886, 2353	<code>\tabu@ignorehfil</code>	2065, 2071, 2077, 2094, 2100, 2106, <u>2142</u>
<code>\tabu@fbox</code>	628, 2210	<code>\tabu@immediate</code>	144, 147
<code>\tabu@finalstrut</code>	<u>139</u> , 1029, 1262	<code>\tabu@indent</code>	<u>601</u>
<code>\tabu@firsthline</code>	617, <u>1662</u>	<code>\tabu@init</code>	<u>601</u>
<code>\tabu@firsthlinecorrection</code>	1662, 1663, <u>1711</u>	<code>\tabu@justify</code>	915, 919, 2082, 2084, 2086, 2089
<code>\tabu@firstline</code>	618, <u>1662</u>	<code>\tabu@justifying</code>	627, 2205
<code>\tabu@firstspace</code>	87, 88	<code>\tabu@keeppls</code>	801, 803, 810
<code>\tabu@fix@arrayright</code>	2226, 2251, 2372	<code>\tabu@l@C@d@rs</code>	511, 518, 525, 537
<code>\tabu@flush</code>	2065, 2070, 2071, 2076, 2094, 2099, 2100, 2105, 2115, 2125	<code>\tabu@lasthline</code>	617, <u>1662</u>
<code>\tabu@footnotetext</code>	<u>2152</u>	<code>\tabu@lasthlinecorrection</code>	1664, 1665, <u>1724</u>
<code>\tabu@footnotes</code>	43, 616, 720, 2153, 2155, 2157, 2170, 2177	<code>\tabu@lastline</code>	618, <u>1662</u>
<code>\tabu@footnotetext</code>	622, 2152, 2350	<code>\tabu@lastn@C@p</code>	2033, 2034
<code>\tabu@FV@CheckEnd</code>	1877, 1887	<code>\tabu@lastnoop</code>	2023, 2033, 2041
<code>\tabu@FV@DefineCheckEnd</code>	<u>1873</u>	<code>\tabu@lastspace</code>	88, 89
<code>\tabu@FV@ListProcessLine</code>	<u>1895</u>	<code>\tabu@lasttry</code>	1125
<code>\tabu@G@extra</code>	182, 186, 188, 189, 190, 192	<code>\tabu@l@textwoe</code>	2194, 2199, 2362
<code>\tabu@G@linesep</code>	213, 217, 219, 220, 221, 223	<code>\tabu@LEADERS</code>	500, <u>505</u>
<code>\tabu@GenericError</code>	149, 1118	<code>\tabu@leaders</code>	498, 499, 502, 812, 1771, 1774
<code>\tabu@get@decimal</code>	1826, 1854	<code>\tabu@leaders@G</code>	499, 521
<code>\tabu@get@decimalspace</code>	1829, 1856	<code>\tabu@leadersstyle</code>	433, <u>498</u>
<code>\tabu@getc@l@r</code>	482, 484, 494	<code>\tabu@leads</code>	510, 517, 526, 530, 531, 534
<code>\tabu@getcolor</code>	416, <u>480</u>	<code>\tabu@leavevmodecolor</code>	1997, 2356, 2357, 2358, 2361
<code>\tabu@getdecimal</code>	1820, <u>1833</u>	<code>\tabu@lefttok</code>	2021, 2047, 2049
<code>\tabu@getdecimal@</code>	1811, 1814, 1820	<code>\tabu@linegoalt@rget</code>	574, 575, 579
<code>\tabu@getdecimal@ignorespaces</code>	1811, 1816, 1825	<code>\tabu@linegoaltarget</code>	573, 574
<code>\tabu@getline</code>	266, 272, <u>387</u> , 797, 1765	<code>\tabu@lines</code>	<u>791</u> , 828, 997
<code>\tabu@getparam</code>	405, 406, 462, 475, <u>479</u>	<code>\tabu@lines@</code>	794
<code>\tabu@gettarget</code>	571, 572, 573	<code>\tabu@linesep</code>	197, 199, 206
<code>\tabu@Gextra</code>	163, 188	<code>\tabu@linestyle@</code>	268, 2387, 2403
<code>\tabu@givespace</code>	1320, 1327, 1336	<code>\tabu@longfalse</code>	538
<code>\tabu@Glinesep</code>	194, 219	<code>\tabu@longpream</code>	989, <u>1070</u>
<code>\tabu@gobblespace</code>	<u>60</u> , 566, 1059, 1822	<code>\tabu@longtrial</code>	<u>1202</u>
<code>\tabu@gobbletoken</code>	<u>60</u> , 166, 169, 171, 178, 180, 197, 200, 202, 209, 211	<code>\tabu@longtrue</code>	547
<code>\tabu@gobbleX</code>	<u>60</u> , 1155	<code>\tabu@ls@</code>	276, 279, 697, 754, 796, 806, 1754
<code>\tabu@gobblex</code>	62, 63, 67	<code>\tabu@ls@G</code>	278, 279, 662, 663, 696, 697, 747, 2387
<code>\tabu@Grestore</code>	190, 221, <u>224</u>	<code>\tabu@ls@Gsave</code>	662, 696
<code>\tabu@Gsave</code>	182, 186, 213, 217, <u>224</u>	<code>\tabu@ls@L</code>	275, 276, 663, 742, 747, 754, 1508
<code>\tabu@halignto</code>	571, 572, 582, 583, 866, 1041, 1255, 1268	<code>\tabu@LT@startpbox</code>	552, <u>2279</u>
<code>\tabu@header</code>	1349, 1413, 1439, 1477	<code>\tabu@ltx@verb</code>	1871, 1872
<code>\tabu@hfil</code>	2126, 2144, 2148	<code>\tabu@maybesiunitx</code>	<u>155</u> , 2055, 2338
<code>\tabu@hfill</code>	2127, 2149	<code>\tabu@measuring</code>	1184, 1199, <u>1202</u>
<code>\tabu@hfuzz</code>	616, 1137, 1334, 1354, 1357, 1418, 1441	<code>\tabu@measuringfalse</code>	615, 995, 1341, 1359, 1950
<code>\tabu@highprioritycolumns</code>	785, 787, 1001	<code>\tabu@measuringtrue</code>	865, 1344, 1523
<code>\tabu@hleads</code>	45, 510	<code>\tabu@message</code>	871, 872, 1329, 1333, 1349, 1365, 1378, 1380, 1385, 1443, 1512, 2317, 2321
<code>\tabu@hline</code>	619, 1666, 1670	<code>\tabu@message@arith</code>	1333, <u>1413</u>
<code>\tabu@hlineAZ</code>	1662, 1663, 1664, 1665, <u>1681</u>	<code>\tabu@message@defaulttarget</code>	871, <u>1401</u>
<code>\tabu@hlineAZsurround</code>	1683, 1685, 1691	<code>\tabu@message@endpboxmeasure</code>	1546, 1561, 1566
<code>\tabu@hlinecorrection</code>	1682, 1700	<code>\tabu@message@etime</code>	1187, <u>1442</u> , 2400
<code>\tabu@hlinescan</code>	1687, 1692	<code>\tabu@message@negcoef</code>	<u>1430</u>
		<code>\tabu@message@reached</code>	1365, <u>1439</u>
		<code>\tabu@message@save</code>	728, <u>1466</u> , 1944, 1947, 2318

<code>\tabu@message@spreadarith</code> . . . . .	1378, <a href="#">1422</a>	<code>\tabu@prepnext@tokORI</code> . . . . .	630, 2031
<code>\tabu@message@target</code> . . . . .	872, <a href="#">1411</a>	<code>\tabu@prev</code> . . . . .	1010, 1016, 1019
<code>\tabu@message@verticalsp</code> . . . . .	<a href="#">1449</a> , 1602, 1614	<code>\tabu@printdecimal</code> . . . . .	1857, 1860, 1866
<code>\tabu@mkarstrut</code> . . . . .	119, 684, 715, 2307, 2310	<code>\tabu@private@columntype</code> . . . . .	772, 779, 782
<code>\tabu@mkarstrut@save</code> . . . . .	684, 715	<code>\tabu@privatecolumns</code> . . . . .	780, 783, 830, 1009
<code>\tabu@mkpreamblebuffer</code> . . . . .	808, 835, 995, 1031, 1037, 1053, 1071, 1083, 1113, 1123, 2037	<code>\tabu@privatecolumntype</code> . . . . .	<a href="#">771</a> , 788, 792, 845, 973, 981
<code>\tabu@modulo</code> . . . . .	<a href="#">60</a> , 351, 368	<code>\tabu@pt</code> . . . . .	1434, 1436, 1467, 1468
<code>\tabu@msg@align</code> . . . . .	1469, 1470, 1472, 1486	<code>\tabu@pushbegins</code> . . . . .	1150, 1167
<code>\tabu@msgalign</code> . . . . .	1350, 1351, 1352, 1355, 1414, 1415, 1416, 1419, 1423, 1424, 1425, 1426, 1427, 1453, 1455, 1458, 1463, 1469	<code>\tabu@quick</code> . . . . .	1569, 1572, 1574, 1582
<code>\tabu@msgalign@PT</code> . . . . .	1353, 1417, 1470	<code>\tabu@quickend</code> . . . . .	1179, <a href="#">1275</a>
<code>\tabu@multicolumn</code> . . . . .	631, 836	<code>\tabu@quickrule</code> . . . . .	596, 600, 1276
<code>\tabu@multispan</code> . . . . .	<a href="#">1651</a> , 1745, 1746, 1792, 1793, 1796	<code>\tabu@RaggedLeft</code> . . . . .	626, 2203
<code>\tabu@n@Gextra</code> . . . . .	164, 189	<code>\tabu@raggedleft</code> . . . . .	624, 2197
<code>\tabu@n@Glinesep</code> . . . . .	195, 220	<code>\tabu@RaggedRight</code> . . . . .	626, 2204
<code>\tabu@naturalX</code> . . . . .	<a href="#">30</a> , 681, 712, 1193, 1549, 1551, 1552	<code>\tabu@raggedright</code> . . . . .	624, 2198
<code>\tabu@naturalX@save</code> . . . . .	681, 712	<code>\tabu@raggedtwoe</code> . . . . .	2200, 2206, 2364
<code>\tabu@naturalXmax</code> . . . . .	683, 713, 1191, 1192, 1371, 1373, 1425, 1551, 1552, 2394	<code>\tabu@rc@</code> . . . . .	249, 338, 339, 347, 355, 360, 361, 367, 372, 373, 378, 699, 755
<code>\tabu@naturalXmax@save</code> . . . . .	683, 713	<code>\tabu@rc@G</code> . . . . .	339, 373, 664, 665, 698, 699, 748, 2386
<code>\tabu@naturalXmin</code> . . . . .	682, 714, 1192, 1372, 1373, 1426, 1553, 1554, 2393	<code>\tabu@rc@Gsave</code> . . . . .	664, 698
<code>\tabu@naturalXmin@save</code> . . . . .	682, 714	<code>\tabu@rc@L</code> . . . . .	338, 361, 665, 748, 755
<code>\tabu@nbc@cols</code> . . . . .	<a href="#">21</a> , 1226, 1495, 1651, 1797, 1801, 1802, 1803, 1935, 2011, 2016, 2037	<code>\tabu@rearstrut</code> . . . . .	<a href="#">109</a> , 253, 688, 1622, 1653
<code>\tabu@NC@list</code> . . . . .	<a href="#">609</a> , 831, 1002, 1003, 1017, 1072, 1084	<code>\tabu@removehfil</code> . . . . .	2064, 2070, 2076, 2093, 2099, 2105, <a href="#">2126</a>
<code>\tabu@negcoeffalse</code> . . . . .	543, 553	<code>\tabu@rescan</code> . . . . .	<a href="#">58</a> , 842, 1209, 1218, 1239
<code>\tabu@negcoeftrue</code> . . . . .	954, 1143	<code>\tabu@reset</code> . . . . .	679, <a href="#">727</a> , 823, 1227, 1622
<code>\tabu@nested</code> <a href="#">21</a> , 349, 352, 353, 354, 363, 364, 365, 366, 369, 370, 371, 602, 717, 719, 1032, 1033, 1043, 1092, 1175, 1176, 1258, 1271, 1290, 1402, 1410, 1433, 1478, 1559, 1560, 1627, 1628, 1629, 1630, 1957, 1964		<code>\tabu@resetls</code> . . . . .	824, 825
<code>\tabu@nestedmeasure</code> . . . . .	581, 591, 599	<code>\tabu@restored</code> . . . . .	608, 1912, 1913
<code>\tabu@new@columntype</code> . . . . .	762, 767, 770, 781	<code>\tabu@restoreeverycr</code> . . . . .	1231, 1232
<code>\tabu@newcolumntype</code> <a href="#">761</a> , 826, 988, 1012, 1015		<code>\tabu@rewritefirst</code> . . . . .	589, 609, <a href="#">988</a>
<code>\tabu@newlinestyle</code> . . . . .	264, 266, 271	<code>\tabu@rewritelast</code> . . . . .	1007, <a href="#">1012</a>
<code>\tabu@nexthlineAZ</code> . . . . .	<a href="#">1681</a>	<code>\tabu@rewritemiddle</code> . . . . .	1006, 1010, <a href="#">1012</a>
<code>\tabu@nocolor</code> . . . . .	<a href="#">153</a> , 506, 1121, 2360	<code>\tabu@rewritemulticolumn</code> . . . . .	<a href="#">826</a>
<code>\tabu@noeverycr</code> . . . . .	1229, 1231	<code>\tabu@rewritevline</code> . . . . .	788, 789, 792
<code>\tabu@nohfil</code> . . . . .	2142, 2143, 2145, 2147	<code>\tabu@rewriteX</code> . . . . .	845, <a href="#">863</a> , 977
<code>\tabu@nolongtabu</code> . . . . .	556, 2346	<code>\tabu@rewriteX@Ss</code> . . . . .	848, 850, 859
<code>\tabu@normalcrbackslash</code> . . . . .	2207, 2209	<code>\tabu@rewriteXrestore</code> . . . . .	<a href="#">875</a> , 977
<code>\tabu@norowcolor</code> . . . . .	<a href="#">153</a> , 1121	<code>\tabu@rewritten</code> . . . . .	790, 799, 800, 846, 858, 876, 881, 883, 935, 937, 953, 959, 961, 963
<code>\tabu@nosiunitx</code> . . . . .	846, 849, 2339	<code>\tabu@Rextra</code> . . . . .	188, 189, 190
<code>\tabu@nowrite</code> . . . . .	<a href="#">142</a> , 1118	<code>\tabu@righttok</code> . . . . .	2019, 2027, 2042, 2046
<code>\tabu@noxfootnote</code> . . . . .	152	<code>\tabu@Rlinesep</code> . . . . .	219, 220, 221
<code>\tabu@noxfootnotes</code> . . . . .	<a href="#">142</a>	<code>\tabu@row@font</code> . . . . .	1982, 1983
<code>\tabu@off</code> . . . . .	<a href="#">30</a> , 402, 420, 422, 423, 424, 435, 488	<code>\tabu@rowc@lors</code> . . . . .	327, 328
<code>\tabu@offiii</code> . . . . .	<a href="#">445</a>	<code>\tabu@rowcolors</code> . . . . .	326, 327, 335
<code>\tabu@offxiii</code> . . . . .	467, 470, 476	<code>\tabu@rowcolorseries</code> . . . . .	334, 336, 377
<code>\tabu@ofxiii</code> . . . . .	<a href="#">445</a>	<code>\tabu@rowcolorserieserror</code> . . . . .	342, 379, 382
<code>\tabu@on</code> . . . . .	<a href="#">30</a> , 402, 420, 421, 422, 424, 434	<code>\tabu@rowfont</code> . . . . .	627, 1981
<code>\tabu@onxiii</code> . . . . .	<a href="#">445</a>	<code>\tabu@rowfont@reset</code> . . . . .	1991, <a href="#">1998</a>
<code>\tabu@oXIII</code> . . . . .	410, 446	<code>\tabu@rowfontreset</code> . . . . .	248, 676, 678, 710, 1989, 1991, 1999, 2006, 2395
<code>\tabu@oxiii</code> . . . . .	446, 449	<code>\tabu@rowfontreset@save</code> . . . . .	676, 710
<code>\tabu@pdftimer</code> . . . . .	2319, 2399, 2400	<code>\tabu@rule@arc@</code> . . . . .	308, 309, 325
<code>\tabu@pream</code> . . . . .	990, <a href="#">1070</a>	<code>\tabu@rule@drsc@</code> . . . . .	290, 291, 295, 296, 307
<code>\tabu@preamble</code> . . . . .	1927, 1942, 1960, 1961	<code>\tabu@rulearc</code> . . . . .	287, 306, 308
<code>\tabu@prepnext@tok</code> . . . . .	630, <a href="#">2007</a>	<code>\tabu@rulecolor</code> . . . . .	283, 284, 288
		<code>\tabu@ruledrsc</code> . . . . .	285, 289
		<code>\tabu@ruledrsc@</code> . . . . .	294, 295
		<code>\tabu@rulesstyle</code> . . . . .	436, <a href="#">498</a> , 1489
		<code>\tabu@sanitizearg</code> <a href="#">93</a> , 267, 343, 344, 391, 1909	
		<code>\tabu@save</code> . . . . .	1914, 1918, 1945
		<code>\tabu@save@decl</code> . . . . .	<a href="#">134</a>
		<code>\tabu@savecounters</code> . . . . .	<a href="#">1109</a> , 1130, 1133, 1207, 1208, 1216, 1217

<code>\tabu@samed@</code> . . . . .	744, 1919	
<code>\tabu@samedec</code> . . . . .	136, 138, 829, 1008	
<code>\tabu@samedparams</code> . . . . .		
. . . . .	728, 739, 750, 1492, 1926, 1929, 1937	
<code>\tabu@samedpream</code> . . . . .		
. . . . .	588, 1491, 1923, 1926, 1930, 1938	
<code>\tabu@samedpreamble</code> . . . . .	1074, 1085, 1924	
<code>\tabu@samederr</code> . . . . .	974, 976, 982, 984, 1974	
<code>\tabu@sameds</code> . . . . .	809, 821, 832, 993	
<code>\tabu@samedwarn</code> . . . . .	1910, 1911, 1974	
<code>\tabu@samedwd</code> . . . . .	1921, 1922, 1973	
<code>\tabu@sameddecimal</code> . . . . .		
. . . . .	1812, 1814, 1815, 1822, 1827, 1830	
<code>\tabu@samedselect</code> . . . . .	1080, 1089, 1091	
<code>\tabu@samedsetcleanup</code> . . . . .	613, 649	
<code>\tabu@samedseteverycr</code> . . . . .		
. . . . .	1096, 1101, 1105, 1206, 1215, 1227, 1270	
<code>\tabu@samedsetextra</code> . . . . .	169, 171, 176, 178, 180	
<code>\tabu@samedsetextrasep</code> . . . . .	173, 185, 187	
<code>\tabu@samedsetlinesep</code> . . . . .	204, 216, 218	
<code>\tabu@samedsetreset</code> . . . . .	613, 727	
<code>\tabu@samedsets@p</code> . . . . .	207, 208, 215	
<code>\tabu@samedsets@ve</code> . . . . .	759, 760	
<code>\tabu@samedsetsave</code> . . . . .	740, 741, 742, 759	
<code>\tabu@samedsetsep</code> . . . . .	200, 202, 207, 209, 211	
<code>\tabu@samedsetstrategy</code> . . . . .	1110, 1114	
<code>\tabu@samedsettarget</code> . . . . .	543, 553, 558	
<code>\tabu@samedsetup</code> . . . . .	587, 601	
<code>\tabu@samedshorttrial</code> . . . . .	1205, 1206, 1214	
<code>\tabu@samedsiunitx</code> . . . . .	845, 847, 2339	
<code>\tabu@samedsiunitxerror</code> . . . . .	856, 860, 862	
<code>\tabu@samedsiunitxfalse</code> . . . . .	2340	
<code>\tabu@samedsiunitxtrue</code> . . . . .	2337	
<code>\tabu@samedskipdecimal</code> . . . . .	1815, 1821, 1823, 1855	
<code>\tabu@samedspread</code> . . . . .	568, 572	
<code>\tabu@samedspread@arith</code> . . . . .	1371, 1396, 1400	
<code>\tabu@samedspreadarith</code> . . . . .	1286, 1370	
<code>\tabu@samedspreadfalse</code> . . . . .	543, 553, 1950	
<code>\tabu@samedspreadheader</code> . . . . .	1422, 1481	
<code>\tabu@samedspreadtarget</code> . . . . .	30,	
. . . . .	867, 868, 1039, 1375, 1379, 1384, 1423, 1428	
<code>\tabu@samedspreadtrue</code> . . . . .	572, 593	
<code>\tabu@samedspxi</code> . . . . .	70, 460, 473	
<code>\tabu@samedstart</code> . . . . .	21, 235, 240, 330, 332, 333,	
. . . . .	356, 1737, 1744, 1745, 1799, 1801, 1806, 1807	
<code>\tabu@samedstart@stop</code> . . . . .	1797, 1798, 1808	
<code>\tabu@samedstartpbox</code> . . . . .	610, 632	
<code>\tabu@samedstartpboxmeasure</code> . . . . .	1194, 1196, 1517	
<code>\tabu@samedstartpboxquick</code> . . . . .	1197, 1535, 1567	
<code>\tabu@samedstartstop</code> . . . . .	1740, 1797	
<code>\tabu@samedstarttime</code> . . . . .	1444, 1445, 1446, 2399	
<code>\tabu@samedstarttimer</code> . . . . .	615, 2319, 2323	
<code>\tabu@samedstop</code> . . . . .	21, 235, 238, 1738, 1741,	
. . . . .	1746, 1800, 1801, 1802, 1803, 1804, 1807	
<code>\tabu@samedstoptime</code> . . . . .	1442, 1444, 1445, 1446	
<code>\tabu@samedstrategy</code> . . . . .	1128, 1171, 1299	
<code>\tabu@samedstrippt</code> . . . . .	957, 1467, 1470, 1487	
<code>\tabu@samedstrtrim</code> . . . . .	80, 97, 289, 485	
<code>\tabu@samedtabarray</code> . . . . .	612, 634	
<code>\tabu@samedtabu@</code> . . . . .	585, 587, 590	
<code>\tabu@samedtabudecimal</code> . . . . .	625, 1809, 1813, 1832	
<code>\tabu@samedtabular</code> . . . . .	540, 545	
<code>\tabu@samedtarget</code> . . . . .	30,	
. . . . .	559, 573, 582, 583, 592, 600, 603, 867,	
. . . . .	869, 870, 1032, 1034, 1038, 1041, 1048,	
. . . . .	1073, 1134, 1137, 1144, 1175, 1252, 1276,	
. . . . .	1296, 1309, 1314, 1332, 1335, 1348, 1352,	
. . . . .	1368, 1375, 1376, 1379, 1381, 1384, 1385,	
. . . . .	1409, 1412, 1416, 1427, 1434, 1438, 1493,	
. . . . .	1525, 1526, 1528, 1556, 1933, 1952, 1954	
<code>\tabu@samedtemp</code> . . . . .	155, 159, 170, 172, 174, 176,	
. . . . .	179, 181, 183, 201, 203, 205, 207, 210,	
. . . . .	212, 214, 285, 286, 289, 290, 291, 293,	
. . . . .	343, 357, 363, 391, 398, 405, 406, 411,	
. . . . .	485, 486, 487, 491, 574, 576, 851, 852,	
. . . . .	883, 939, 942, 957, 958, 961, 1013, 1016,	
. . . . .	1019, 1026, 1049, 1054, 1057, 1058, 1061,	
. . . . .	1062, 1063, 1068, 1158, 1159, 1160, 1161,	
. . . . .	1513, 1518, 1519, 1521, 1525, 1526, 1543,	
. . . . .	1550, 1562, 1591, 1594, 1651, 1652, 1669,	
. . . . .	1672, 1682, 1689, 1690, 1697, 1698, 1699,	
. . . . .	1709, 1751, 1780, 1814, 1817, 1818, 1834,	
. . . . .	1835, 1836, 1837, 1838, 1839, 1840, 1841,	
. . . . .	1842, 1843, 1844, 1845, 1846, 1847, 1848,	
. . . . .	1849, 1850, 1851, 1852, 1861, 1862, 1863,	
. . . . .	1909, 1910, 1911, 1913, 1948, 2195, 2196,	
. . . . .	2197, 2198, 2201, 2202, 2203, 2204, 2205	
<code>\tabu@samedtextbar</code> . . . . .	100, 283, 828, 997	
<code>\tabu@samedthebody</code> . . . . .	43, 1076, 1077, 1079,	
. . . . .	1087, 1088, 1152, 1211, 1220, 1241, 1245	
<code>\tabu@samedthehleaders</code> . . . . .	512, 1775	
<code>\tabu@samedthehline</code> . . . . .	1747, 1773, 1778	
<code>\tabu@samedthehrule</code> . . . . .	503, 1772, 1776	
<code>\tabu@samedtheleaders</code> . . . . .	512, 519, 532, 536	
<code>\tabu@samedtheparam</code> . . . . .	482, 488	
<code>\tabu@samedtherule</code> . . . . .	524, 527, 528, 529, 531, 533, 535	
<code>\tabu@samedthestyle</code> . . . . .	270, 275, 278, 392, 394, 395, 397,	
. . . . .	431, 432, 441, 796, 800, 801, 1742, 1754, 1771	
<code>\tabu@samedthetarget</code> . . . . .	608, 644, 870, 1404, 1405	
<code>\tabu@samedthevleaders</code> . . . . .	519, 814	
<code>\tabu@samedthevline</code> . . . . .	799, 811, 820, 1645, 1650, 1659	
<code>\tabu@samedthevrule</code> . . . . .	503, 818	
<code>\tabu@samedthick</code> . . . . .	30, 402, 428, 429,	
. . . . .	433, 437, 438, 1692, 1693, 1705, 1706, 1707	
<code>\tabu@samedtitles</code> . . . . .	1329, 1478	
<code>\tabu@samedto</code> . . . . .	567, 571	
<code>\tabu@samedtr@cing</code> . . . . .	2300, 2302, 2325	
<code>\tabu@samedtracing</code> . . . . .	2297, 2298, 2299, 2301	
<code>\tabu@samedtracing@save</code> . . . . .	2318, 2322	
<code>\tabu@samedtracinglines</code> . . . . .	1749, 2303, 2304	
<code>\tabu@samedtrial</code> . . . . .	1202	
<code>\tabu@samedtrialh@k</code> . . . . .	140, 141, 1115	
<code>\tabu@samedtrimspaces</code> . . . . .	84, 87, 91	
<code>\tabu@samedtrivlist</code> . . . . .	621, 2208	
<code>\tabu@samedusetabu</code> . . . . .	1047,	
. . . . .	1073, 1514, 1931, 1941, 1947, 1951, 1953	
<code>\tabu@samedverb</code> . . . . .	1868, 1871	
<code>\tabu@samedverbatim</code> . . . . .	59, 1867	
<code>\tabu@samedverticaldynamicadjustment</code> . . . . .	1260, 1631	
<code>\tabu@samedverticalinit</code> . . . . .	1099, 1182, 1264, 1621	
<code>\tabu@samedverticalmeasure</code> . . . . .	1099, 1182, 1265, 1585	
<code>\tabu@samedverticalsp@lcr</code> . . . . .	1588, 1598	
<code>\tabu@samedverticalsp@pmb</code> . . . . .	1595, 1607	
<code>\tabu@samedverticalspacing</code> . . . . .	1100, 1183, 1266, 1588, 1595	
<code>\tabu@samedvleeds</code> . . . . .	45, 517	
<code>\tabu@samedvlinearg</code> . . . . .	789, 795	
<code>\tabu@samedwarn</code> . . . . .	151	
<code>\tabu@samedwarn@cellspace</code> . . . . .	2347, 2375, 2381	
<code>\tabu@samedwarn@usetabu</code> . . . . .	1952, 1970	
<code>\tabu@samedwarncolour</code> . . . . .	491, 495, 497	



<code>\tabu@wd</code> . . . . .	967, 968, 1309, 1313, 1314, 1398, 1432, 1544, 1560, 1564, 1565, 1973
<code>\tabu@wddef</code> . . . . .	1146, 1322, 1324, 1387, 1510, 1545, 1558, 1921, 1973
<code>\tabu@WRITE</code> . . . . .	145, 148, 150, 151
<code>\tabu@write</code> . . . . .	143, 147
<code>\tabu@X</code> . . . . .	68, 1161
<code>\tabu@Xalign</code> . . . . .	912, 913, 914, 915, 916, 917, 918, 919, 927, 933
<code>\tabu@Xarg</code> . . . . .	863, <u>877</u>
<code>\tabu@Xc@ef</code> . . . . .	940, 942, 944
<code>\tabu@xcline</code> . . . . .	1751, 1779, 1788
<code>\tabu@Xcoef</code> . . . . .	921, 923, 939, 941
<code>\tabu@Xcoefs</code> . . . . .	958, 993, 1040, 1042, 1135, 1303, 1336, 1371, 1921
<code>\tabu@Xcol</code> . . . . .	<u>21</u> , 878, 957, 994, 1141, 1524, 1544, 1545, 1562, 1564, 1565
<code>\tabu@Xd@sp</code> . . . . .	945, 946, 951
<code>\tabu@Xdisp</code> . . . . .	879, 947, 960
<code>\tabu@xfootnote</code> . . . . .	623, 1120, 2159, 2351
<code>\tabu@xfootnotetext</code> . . . . .	622, 2156
<code>\tabu@xhline</code> . . . . .	619, 1669, 1671, 1680
<code>\tabu@xhlineAZ</code> . . . . .	<u>1681</u>
<code>\tabu@Xinit</code> . . . . .	1135, 1141, 1148
<code>\tabu@Xlcr</code> . . . . .	878, 928, 930, 931, 960
<code>\tabu@Xm@th</code> . . . . .	922, 945
<code>\tabu@Xmath</code> . . . . .	879, 946, 960, 962
<code>\tabu@Xparse</code> . . . . .	<u>877</u>
<code>\tabu@Xparsespace</code> . . . . .	889, 920
<code>\tabu@Xrewritten</code> . . . . .	886, <u>952</u>
<code>\tabu@Xsum</code> <u>38</u> , 1135, 1141, 1322, 1343, 1419	
<code>\tabu@Xtest</code> . . . . .	888, 890, 926
<code>\tabu@Xtype</code> . . . . .	909, 910, 911, 934, 938
<code>\tabucline</code> . . . . .	<u>16</u> , 1686, 1695, 1699, <u>1737</u>
<code>\tabucline@scan</code> . . . . .	1686, 1694
<code>\tabucolumn</code> . . . . .	<u>784</u>
<code>\tabucolX</code> . . . . .	<u>30</u> , 881, 883, 942, 954, 955, 957, 967, 968, 970, 1033, 1134, 1135, 1138, 1145, 1176, 1251, 1295, 1307, 1309, 1311, 1340, 1343, 1350, 1361, 1367, 1376, 1392, 1414, 1419, 1434, 1436, 1494, 1525, 1526, 1563, 1934
<code>\tabucolX@init</code> . . . . .	1124, 1134, 1140, 1389
<code>\tabudecimal</code> . . . . .	<u>21</u> , 625, <u>1809</u>
<code>\tabefaulttarget</code> . . . . .	606, 608, 639, 645, 1119, 1403, 1406, 2292, 2406
<code>\tabuDisableCommands</code> . . . . .	<u>140</u> , 378
<code>\tabular</code> . . . . .	545
<code>\tabulineoff</code> . . . . .	423, 2396
<code>\tabulineon</code> . . . . .	421, 2396
<code>\tabulinesep</code> . . . . .	<u>12</u> , <u>193</u> , 384, 2379
<code>\tabulinestyle</code> . . . . .	<u>16</u> , <u>272</u> , 385, 2404
<code>\tabuphantomline</code> . . . . .	<u>12</u> , <u>1638</u>
<code>\tabureset</code> . . . . .	<u>18</u> , <u>383</u>
<code>\taburow</code> . . . . .	<u>21</u>
<code>\taburowcolors</code> . . . . .	<u>18</u> , 249, <u>326</u> , 385, 2405
<code>\taburulecolor</code> . . . . .	<u>16</u> , <u>283</u> , 385
<code>\tabuscantokensfalse</code> . . . . .	1092, 2382
<code>\tabuscantokenstrue</code> . . . . .	546, 555, 2294
<code>\tabustrutrue</code> . . . . .	<u>39</u> , 122, 128, 132, 1271, 1602, 1614, 1717, 1733, 2308, 2311
<code>\tabuthepreamble</code> . . . . .	1079, 1088, 1097, 1102, 1106, 1210, 1219, 1240, 1244
<code>\thempfn</code> . . . . .	2164, 2173
<code>\thetaburow</code> . . . . .	<u>21</u>
<code>\toks</code> . . . . .	134, 225, 228, 229, 231, 1919, 1922, 1923, 1924, 1929, 1930, 1932, 1937, 1938, 1943, 2038, 2044, 2047, 2048
<code>\tracingtabu</code> . . . . .	<u>24</u> , <u>2295</u>
<b>U</b>	
<code>\unhbox</code> . . . . .	509, 1222
<code>\unkern</code> . . . . .	2131
<code>\unskip</code> . . . . .	139, 2136, 2269
<code>\unvbox</code> . . . . .	516
<code>\usetabu</code> <u>19</u> , 739, 973, 980, 1004, 1949, 1971, 1978	
<code>\usetabu_(private_column_type)</code> . . . . .	<u>973</u>
<b>V</b>	
<code>\value</code> . . . . .	2163, 2172
<code>\vbox</code> . . . . .	516, 1593, 1759
<code>\vcenter</code> . . . . .	1592
<code>\vskip</code> . . . . .	517, 1673, 1674, 1718, 1728, 1782
<code>\vss</code> . . . . .	517
<code>\vtop</code> . . . . .	813, 1517, 1556, 1591, 1760
<b>W</b>	
<code>\write</code> . . . . .	143, 146, 1118
<b>X</b>	
<code>\X_(private_column_type)</code> . . . . .	<u>845</u>
<code>\xglobal</code> . . . . .	353, 365, 370
<code>\xleaders</code> . . . . .	534, 1764, 1769